Universidad Carlos III de Madrid

Escuela Politécnica Superior

Grado en Ingeniería Telemática

Mecanismos de optimización cross-layer para el tráfico de vídeo en redes 4G

Trabajo Fin de Grado

Autor:	**Carlos A. Donato Morales**
Tutor:	**Carlos Jesús Bernardos Cano**
Director:	**Pablo Serrano Yáñez-Mingot**

Junio de 2014

A mis padres, a mis hermanos y a mis abuelos

Agradecimientos

En primer lugar, agradecer a mis abuelos, padres y hermanos el apoyo y la paciencia infinita que me han mostrado.

Agradecer a mis tutores, Carlos J y Pablo, la confianza depositada en mí, el tiempo que me han dedicado y su ayuda. Sin ellos, no habría podido vivir dos grandes experiencias de mi vida ni habría descubierto mi vocación.

Dar las gracias a Antonio, sin su supervisión y motivación, este Trabajo de Fin de Grado no habría salido adelante.

A Alba, fuente constante de inspiración y apoyo incondicional. Nunca habría llegado hasta aquí sin ella.

A mi gran amigo Sergio. Sus consejos y tiempo sólo se puede pagar con una gran amistad. También a mis amigos de toda la vida, nunca han faltado cuando los he necesitado.

Para terminar, a mis compañeros y amigos Alberto Díaz por su incalculable ayuda y aventuras; Alberto Martín, por las andanzas vividas junto al anterior; Javier de Benito, compañero inigualable y gran amigo; Jorge por el plus que hacía falta al equipo en los momentos de tirar la toalla.

No guardes nunca en la cabeza aquello que te quepa en un bolsillo.
Albert Einstein (1879-1955)

Resumen

Los operadores y proveedores de servicios de todo el mundo han observado un incremento significativo de vídeo en redes móviles así como un crecimiento en la demanda de mayores tasas de transferencia sobre redes inalámbricas que, en muchos casos, sobrepasan las ofertadas por tecnologías de última generación como el LTE. Comparado con el núcleo de la red móvil, las congestiones ocurren de manera mas frecuente en los interfaces de red inalámbricos por su naturaleza. La calidad de vídeo se ve dramáticamente degradada debido a las congestiones y las variaciones de las condiciones del canal en las celdas, disminuyendo la *Quality-of-Experience* (QoE) de los usuarios.

El proyecto europeo MultimEDia transport for mobIlE Video AppLications (MEDIEVAL) tiene como objetivo evolucionar la actual arquitectura de Internet móvil centrándose en la optimización de los servicios de vídeo. Al combinar aspectos de movilidad en la red CDN y mediante el diseño de las técnicas adecuadas de ingeniería de tráfico, trata de responder a las limitaciones de la QoE en flujos de vídeo.

Esta arquitectura funciona sobre un sistema basado *cross-layer* junto a una codificación de vídeo en capas, obteniendo así un uso mas eficiente del tráfico de vídeo en redes móviles. Esta estrategia de gestión de tráfico se centra en la QoE en lugar de los parámetros tradicionales de la *Quality-of-Service* (QoS). Las mejoras específicas de vídeo se introducen en las diferentes capas de la pila de protocolos. Por otra parte, el enfoque de *cross-layer* proporciona un soporte mejor de vídeo a un coste menor que las soluciones convencionales.

En este Trabajo Fin de Grado se describe la solución diseñada para proveer un servicio mejorado de vídeo sobre redes inalámbricas enfocado a la QoE del usuario, así como el montaje de una demostración que presenta la funcionalidad.

Palabras clave: IPv6, LTE, Wi-Fi, redes móviles, video, optimización, streaming, redes celulares, SVC.

Abstract

Operators and service providers around the world have observed a significant increase of video on mobile networks as well as an increase in the demand for higher transfer rates over wireless networks, in many cases, exceed those offered by latest technologies like LTE. Compared to the core of the mobile network congestions occur more frequently in the wireless network interfaces by its physical nature. The video quality is dramatically degraded due to congestion and variations in channel conditions in the cells, decreasing the Quality-of-Experience (QoE) of users.

The European project MultimEDIa transport for mobile Video AppLications (MEDIEVAL) aims to evolve the current architecture of mobile internet focusing on the optimization of video services. By combining aspects of mobility in the CDN and by designing the proper techniques of traffic engineering, seeks to answer the limitations of QoE in video streams.

This architecture applies a cross-layer framework to efficiently handle video traffic in the mobile network. This traffic management strategy focuses on QoE rather than the traditional parameters of the Quality-of-Service (QoS). Specific video improvements are introduced in different layers of the protocol stack. Moreover, the approach cross-layer video provides a better support at a lower cost than conventional solutions.

In this Bachelor Thesis is described the solution designed to provide an improved video service over wireless networks focused on QoE of user, as well as the installation of a demonstration that shows the solution functionality.

Keywords: IPv6, LTE, Wi-Fi, mobile networks, video, optimization, cellular networks, streaming, SVC.

Índice General

Lista de Figuras

Glosario

La lista de los acrónimos utilizados en este proyecto es la siguiente:

3GPP 3rd Generation Partnership Project

AP Access Point

ARQ Automatic Repeat Request

CDN Content Delivery Network

DMOS Differential Mean Opinion Score

EDCA Enhanced Distributed Channel Access

EPS Evolved Packet System

eMBMS evolved MBMS

FEC Forward Error Correction

GUI Graphical User Interface

HCF Hybrid Coordinator Function

HCCA Hybrid Controlled Channel Access

IEEE Institute of Electrical and Electronics Engineers

IP Internet Protocol

LTE Long Term Evolution

LTS Long Term Support

MAC Medium Access Control

MAR Mobile Access Router

MBMS MulticastIBroadcastMultimedia Service

Mbps Megabit per second

MEDIEVAL MultiMEDia transport for mobIlE Video AppLications

MIHF Mobile Independent Handover Function

MMAS Measurements and Medium Access Strateg

MN Mobile Node

MTU Maximum Transfer Unit

ODTONE Open Dot Twenty One

PDCP Packet Data Convergence Protocol

QoE Quality of Experience

QoS Quality of Service

RTCP Real Time Control Protocol

RTP Real-time Transport Protocol

RTSP Real Time Streaming Protocol

SAP Service Access Point

SIP Session Initiation Protocol

SSID Service Set IDentifier

SSL Secure Sockets Layer

SVC Scalable Video Coding

TLS Transport Layer Security

TE Traffic Engineering

TO Transport Optimization

UC3M Universidad Carlos III de Madrid

UDP User Datagram Protocol

UMTS Universal Mobile Telecommunication System

VFSIC Video Frames Selection and Interface Configuration

VPN Virtual Private Network

VS Video Services

VQM Video Quality Metric

WA Wireless Access

Wi-Fi Wireless Fidelity

WLAN Wireless Local Area Network

WLANMM WLAN Monitoring Module

WMDC Wlan Monitoring Dynamic Configuration

XLO Cross-Layer Optimizer

Parte I

Introducción

Capítulo 1

Introducción

1.1. Introducción

El *streaming* de vídeo está considerado como una de las aplicaciones más importantes y desafiantes para las redes celulares de última generación. Las infraestructuras actuales no están preparadas para hacer frente a la creciente cantidad de tráfico de vídeo. La arquitectura de Internet actual, y en particular la Internet móvil, no fue diseñada con los requisitos de vídeo en mente y, como consecuencia, su arquitectura es muy ineficiente a la hora de manejar tráfico de video.

En este trabajo se describe el desarrollo de una solución para optimizar el tráfico de vídeo sobre redes móviles así como el montaje de dicha solución sobre un banco de pruebas. Dicha solución se enmarca dentro del proyecto MEDIEVAL como la tercera demostración de las tres presentadas.

1.2. Objetivos

- Despligue y configuración del testbed ubicado en el Departamento de Ingeniería Telemática de la Universidad Carlos III de Madrid.
- Configuración e integración de las herramientas necesarias para la demostración.
- Desarrollo de la demostración en el testbed ubicado en el Departamento de Ingeniería Telemática de la Universidad Carlos III de Madrid.
- Pruebas y análisis de los resultados.
- Migrar las configuraciones y módulos al testbed remoto.
- Repetir las pruebas y analizar los resultado en el testbed remoto.

1.3. Fases del desarrollo

El Trabajo Fin de Grado se dividió en las fases de desarrollo siguientes:

- **Documentación y análisis del proyecto europeo MEDIEVAL**: estudio del proyecto MEDIEVAL y análisis de la tercera demostración.
- **Despliegue de la red**: montaje del testbed en el Departamento de Ingeniería Telemática de la Universidad Carlos III de Madrid.
- **Instalación y configuración del banco de pruebas**: probar las herramientas existentes que incluyeron en la demostración.
- **Desarrollo e integración de los módulos**: implementación de los módulos necesarios para tener del prototipo con la funcionalidad requerida.
- **Pruebas de validación**: se realizaron múltiples pruebas para comprobar el comportamiento y funcionamiento de los distintos módulos era el esperado.
- **Pruebas y medidas**: se recrearon los diferentes escenarios de la demostración y se obtuvieron medidas para verificar si el comportamiento era el previsto.
- **Replicar la demostración en el banco de pruebas remoto**: migrar todas las configuraciones y desarrollos al testbed remoto y comprobar que dicha réplica funcionaba igual que la original en el Departamento de Ingeniería Telemática de la Universidad Carlos III de Madrid.
- **Pruebas y medidas en el testbed remoto**: con la demostración ya montada en los laboratorios ubicados en Sophia Antípolis, se procedió a repetir todas las pruebas realizadas en nuestro banco de pruebas de la UC3M.
- **Evaluación de los resultados**: con los resultados obtenidos, se analizaron y se compararon con las obtenidas en los diferentes bancos de pruebas para validar las pruebas.

1.4. Estructura de la memoria

La memoria se divide en varias partes, que a su vez se dividen en distintos capítulos. El contenido de cada parte y capítulo se resume a continuación:

1. **Primera parte: Introducción**. En esta parte se explican los objetivos, las fases del trabajo y la estructura de la memoria. Sólo contiene un capítulo:
   - Capítulo 1. Introducción
2. **Segunda parte: Estado del arte**. Se explican los diferentes retos que motivaron la creación el proyecto MEDIEVAL
   - Capítulo 2. Tráfico de vídeo en redes móviles
3. **Tercera parte: Trabajo realizado**. En esta parte se explica el trabajo realizado para el desarrollo y configuración de la demostración.
   - Capítulo 3. Arquitectura del prototipo.
   - Capítulo 4. Despliegue del prototipo.
   - Capítulo 5. Escenarios de prueba.

4. **Cuarta parte: Conclusiones y trabajos futuros**. En esta parte se explican las conclusiones obtenidas del trabajo y futuros proyectos para ampliar el actual. Sólo contiene un capítulo:

   - Capítulo 6. Conclusiones y trabajos futuros

5. **Quinta parte: Anexos**. En esta última parte se amplía la información de algunas partes del Trabajo Fin de Grado:

   - Apéndice A: Planificación de tareas y presupuesto. En este anexo se describen las tareas del proyecto y los costes de estas.
   - Apéndice B: Configuración de los nodo móviles. Se presenta la configuración detallada de los diferentes nodos móviles usados en nuestra demostración.
   - Apéndice C: Configuración de los routers de acceso móvil. Se presenta la configuración detallada de los diferentes routers de acceso móviles empleados para proveer servicio en la arquitectura propuesta.
   - Apéndice D: Configuración del servidor de vídeo. Se presenta la configuración detallada del servidor de vídeo que se utiliza para dar servicio multimedia de la demostración.

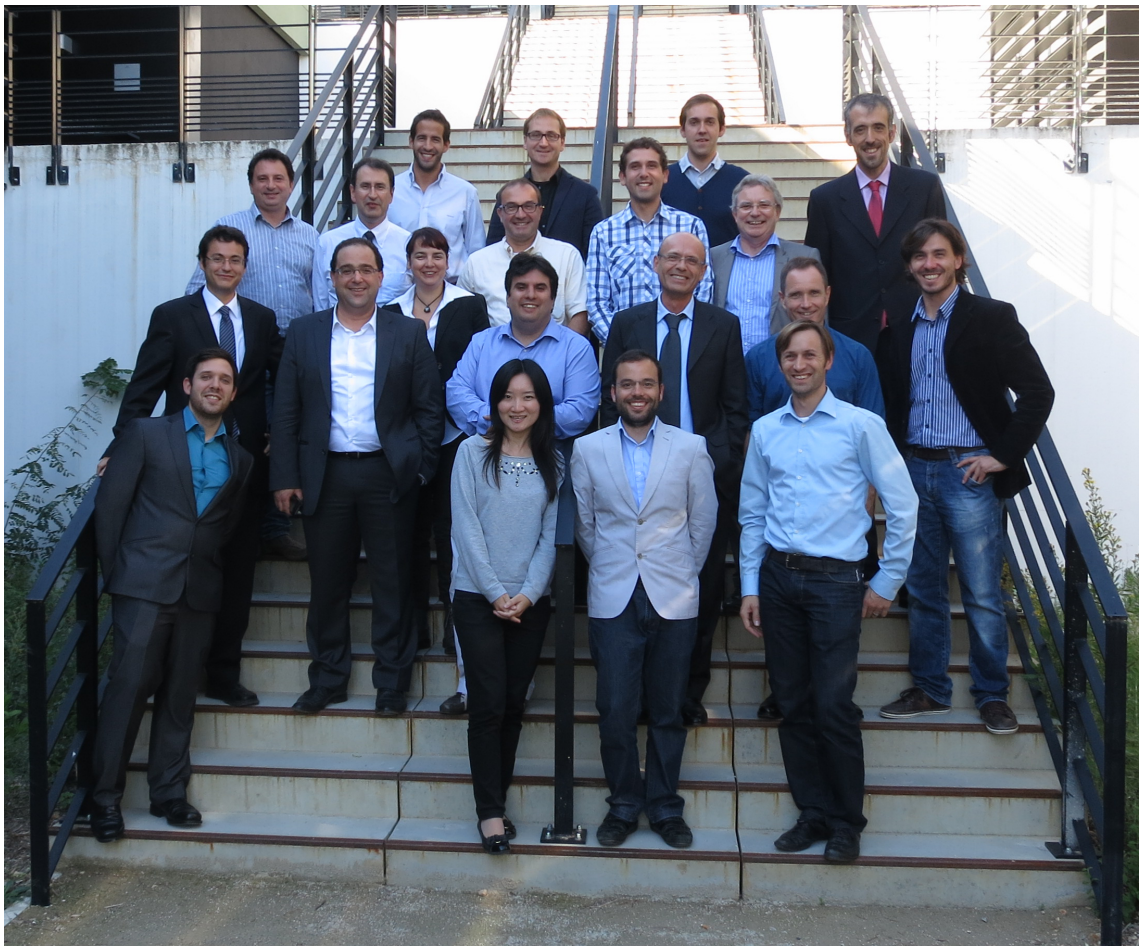

Figura 1.1: Equipo del proyecto europeo MEDIEVAL

Parte II

Estado del Arte

Capítulo 2

Tráfico de vídeo en redes móviles

2.1. Introducción

La llegada de *smartphones* y dispositivos móviles al mercado ha dado un giro a la manera que entendíamos internet tradicionalmente. Esta revolución ha significado tener todo el contenido de la red de redes al alcance de nuestras manos en cualquier lugar y momento. Gracias a esto, el tráfico de internet sobre redes móviles se ha visto incrementado exponencialmente. Si analizamos qué tipo de tráfico ha sufrido más cambios, vemos que el tráfico de vídeo es el que más ha aumentado y aumentará en los próximos años.

Con esto en mente, nació el proyecto europeo MultimEDia transport for mobIlE Video AppLications (MEDIEVAL) una solución para evolucionar la arquitectura actual de red y distribuir de manera más eficiente el tráfico multimedia sobre redes móviles [Ind].

2.2. Motivaciones y retos del proyecto MEDIEVAL

Las diferentes arquitecturas de comunicaciones por radio actuales han mejorado significativamente sus tasas de datos y disponibilidad en los últimos años, a fin de conseguir la transmisión de vídeo efectiva sin cables. Sin embargo, no es suficiente *per se* ya que las tecnologías de última generación son, todavía, carentes de un enfoque sistemático para la gestión de recursos de radio que permita la entrega de vídeo de manera óptima. La mayoría de los diseños de red no proporcionan mecanismos de adaptación a los requisitos de la aplicación dependiendo de las condiciones del canal, ni se tienen en cuenta el acceso inalámbrico en el diseño.

Cuando se exploran las interacciones entre los diferentes niveles de red nos solemos centrar, a menudo, en la capa de aplicación (por ejemplo, cómo impactan los efectos de distorsión debido a la pérdida de paquetes afecta en la calidad percibida por el usuario) en lugar del nivel de acceso o capa física (por ejemplo, la estimación de la capacidad de transmisión de un medio inalámbrico), donde una mejora de la gestión también es clave para lograr una transmisión fiable y fluida. Por lo tanto, existen varias direcciones donde explorar, y en este sentido MEDIEVAL presenta diferentes contribuciones en la gestión de la red. Un resumen de los desafíos y escenarios de esta aplicación se puede encontrar en la Tabla 2.1 al final de esta sección.

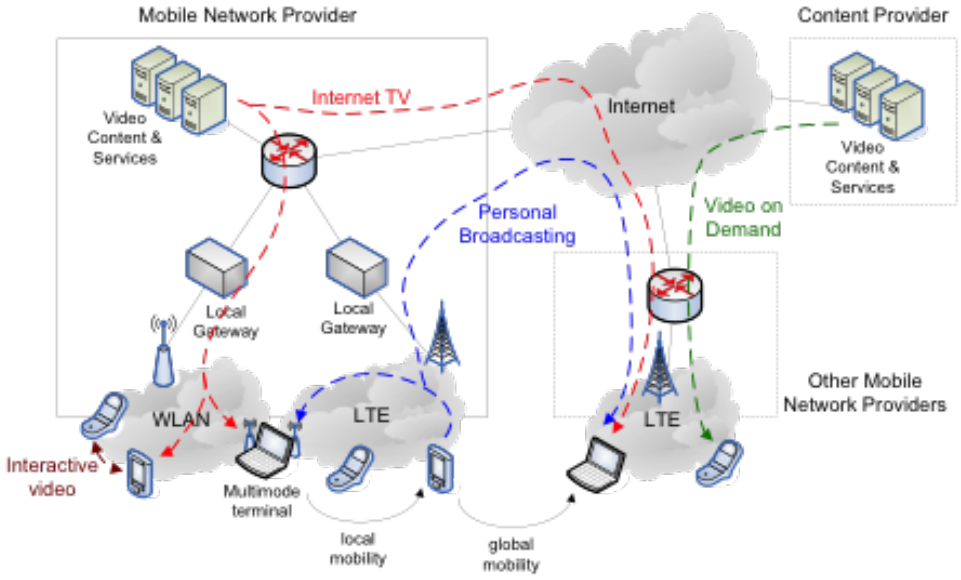

Figura 2.1: Arquitectura de MEDIEVAL. Fuente [FP7]

En primer lugar, el contenido de vídeo es, frecuentemente, heterogéneo y se traduce en flujos con características muy variables, por ejemplo, en términos de ancho de banda y retardo requerido; también se espera que los dispositivos finales sean muy diferentes entre sí, de este modo imponer diferentes restricciones en términos de Quality-of-Experience (QoE) que el usuario final pueda solicitar. Por lo tanto, la adaptación de flujo de vídeo surge como un desafío importante para el estudio dentro del proyecto.

Esto implica tener en cuenta varios elementos al mismo tiempo, tales como el tipo de dispositivo que el usuario final va a emplear, así como las condiciones de la red y del canal que son especialmente importantes en las redes inalámbricas móviles, donde estos elementos suelen ser más variables. A pesar de que se espera que la proporción de los *streaming* de vídeo vaya ser dominante en las redes de uso general, tales como Internet, se pretende coexistir con otros tipos de tráfico, incluso con requisitos mucho más diferentes. Los escenarios donde el tráfico de vídeo compite con el tráfico en tiempo no-real ya están considerados, por ejemplo, en los recientes extensiones de la IEEE 802.11 [ST99].

Hasta cierto punto, los usuarios de vídeo pueden tener una QoE buena si sólo se considera un mecanismo de control y reciben mayor prioridad que los usuarios de datos [SD00]. Sin embargo, este enfoque carece de generalidad y en la actualidad sólo garantiza un buen desempeño en un número de escenarios limitados. Los esquemas recientes proponen una mejora en el rendimiento del estándar y de la configuración defecto sin la necesidad de cambiar la especificación actual.

Estas propuestas, ya sea sobre la base de HCF (Hybrid Coordinador Function) Controlled Channel Access (HCCA) [GB] o Enhanced Distributed Channel Access (EDCA) se dividen de forma adaptativa el tiempo dedicado a tiempo real y no-real del servicio de tiempo. Su principal inconveniente es que son, normalmente, mecanismos basados en la heurística y por lo tanto no garantizan un rendimiento óptimo. Otro elemento de interés para la aplicación de transmisión de vídeo sobre medios inalámbricos, que es inherentemente un medio poco fiable, es la aplicación de paradigmas para el control de errores. Actualmente,

la mayoría de las técnicas se basan en mecanismos de Forward Error Correction (FEC), mientras que, recientemente, se ha mostrado que Automatic Repeat Request (ARQ) puede ser beneficioso, especialmente si se combina con FEC con el fin de realizar una esquema ARQ híbrido [LB].

Por otra parte, la mayoría de los enfoques de control de errores a menudo se enfrentan con la capa de aplicación, y aplican técnicas que son casi completamente independientes de otras estrategias aplicadas en las capas inferiores. Esta limitación de impactos en el uso eficiente de la red no tiene en cuenta el estado de esta para la puesta a punto del mecanismo de control de errores. Con este fin, se ha propuesto emplear estrategias de FEC y ARQ que interactúen entre las diferentes capas de manera *cross-layer*, por ejemplo, permitir técnicas de control de errores en la fase del control de acceso al medio (MAC) también controlado por el receptor final a través del protocolo de señalización Real-Time Transport Control Protocol (RTCP) [MvdS].

En general, la optimización *cross-layer* es un completo desafío y muy relevante para MEDIEVAL. La aplicación exacta de las técnicas *cross-layer* depende de la tecnología de acceso inalámbrica específica que se trate. Sin embargo, es deseable tener un enfoque transparente y general, que siga siendo compatible con el enfoque por capas y, posiblemente, transparente para la aplicación, transporte y movilidad de los elementos. Por otra parte, hemos también de ser conscientes de que la optimización *cross-layer* puede implicar, teóricamente, buenos enfoques que son inaplicables en la práctica; por lo tanto, la cantidad exacta de la señalización necesaria para el intercambio *cross-layer* debe ser evaluado cuidadosamente.

Por estas razones, se identifican dos desafíos como la definición de un interfaz abstracto llamado Layer 2.5 que opera entre el nivel de enlace y el nivel de red, y la necesidad de evaluar la viabilidad práctica de cualquier propuesta técnica de optimización.

Otro reto que implica la rápida expansión de los servicios multimedia es que, a menudo, tienen una amplia difusión dentro de las redes sociales, orientado al uso colectivo. Este es el caso, por ejemplo, de fotografía y de intercambio de vídeo en Facebook, o vía MMS. Sin embargo, hay una falta de mecanismos de interconexión tengan en cuenta esta tendencia. Se siguen echando en falta más estándares para incluir servicios de anuncio a escala y de descubrimiento así como el mapeo de grupos de usuarios de servicios y la gestión de los diferentes contenidos basados en diferentes fuentes.

Por esta razón, es importante tener en cuenta en vídeo como medio de difusión en la red. Un escenario de referencia en este sentido puede ser los MBMS (MulticastIBroadcast Multimedia Services) [V7.06], que es una mejora para el sistema UMTS y proporciona una capacidad de punto a multipunto, minimizando la red y el uso de recursos de radio. Los eMBMS (evolved MBMS) es la adaptación a la arquitectura del sistema de paquetes evolucionado (EPS) y el acceso de LTE.

La mejor manera de utilizar las técnicas anteriores depende de la tecnología considerada en particular. Por ejemplo, en tecnologías basadas en coordinación, el control de admisión puede ser utilizado para limitar la cantidad de tráfico en la red y por lo tanto las situaciones de congestión pueden ser fácilmente evitadas. Por otra parte, en tecnologías basadas en contención que carecen de control de admisión, la congestión se puede producir en mayor medida y se requieren técnicas para reducir al mínimo su impacto en el tráfico de video. Uno de los principales objetivos de MEDIEVAL es mejorar la QoE de vídeo mediante la explotación tanto independiente de la tecnología y también las características dependientes de la tecnología.

Retos
- Adaptación del flujo de vídeo - Coordinación con otros tipos de tráfico - Control de Errores - Optimización de cross-layer - Multicast/Broadcast
Escenarios tecnológicos
- Acceso inalámbrico (LTE) basado en coordinación - Acceso inalámbrico (Wi-Fi) basado en contienda - Interfaz abstracto en Layer 2.5

Tabla 2.1: Resumen de los retos de MEDIEVAL

2.3. Arquitectura de MEDIEVAL

La arquitectura global de MEDIEVAL consiste en cuatro bloques principales, llamados Video Services (VS), Transport Optimization (TO), Mobility Management y Wireless Access (WA). La arquitectura de red de referencia es la de la Third Generation Partnership Project (3GPP) pero incluyendo también las redes no 3GPP, tales como puntos de acceso WLAN para, por ejemplo la *traffic offload*. La figura 2.2 muestra los componentes básicos funcionales. Los componentes clave del modelo propuesto pertinentes a este trabajo se presentan a continuación.

El VS controla la gestión de sesiones, el control de vídeo, y la adaptación de contenido. También marca el flujo de paquetes de vídeo con las prioridades en función de su importancia dada la calidad de vídeo. La identificación de prioridad se realiza en las cabeceras IP y se puede acceder fácilmente por las entidades adaptando el flujo de vídeo en la red central y en el acceso inalámbrico. Así, la inspección de paquetes en la red o en el acceso puede evitarse.

El WA soporta la conectividad a través de tecnologías de acceso heterogéneas, basado en el estándar IEEE 802.21 [fLman08] . Se despliega una función de Media Independent Handover Function (MIHF) que actúa como una interfaz abstracta entre diferentes tecnologías inalámbricas y las capas superiores de vídeo, permitiendo independencia del medio, esquemas de optimización basados en QoE. Así interactúa con las capas de enlace a través de Service Access Points (SAPs), implementando las primitivas y parámetros para la información extraída del flujo de una manera *cross-layer*. La interfaz también se conoce como Layer 2.5 y está extendida para optimizar el control de la movilidad en entornos de servicio de vídeo. En cuanto a la capa de enlace de LTE, los Video Frames Selection and Interface Configuration (VFSIC) selecciona los componentes y da prioridad a los *frames* de vídeo.

A través de la información recibida a través del MIHF, el VFSIC filtra los *frames* de vídeo en el plano del usuario del interfaz radio, permitiendo sólo las adaptaciones de la carga del enlace inalámbrico. El VFSIC trabaja conjuntamente con un módulo de control, Measurements and Medium Access Strategies (MMAS), proporcionando la información acerca de la calidad del enlace entre el equipo del usuario y el eNodeB. El MMAS supervisa la ocupación de la colas del nivel 2 en el eNodeB, siendo por tanto capaz de detectar la congestión en la célula inalámbrica. Para WLAN, estas funcionalidades son proporcionadas

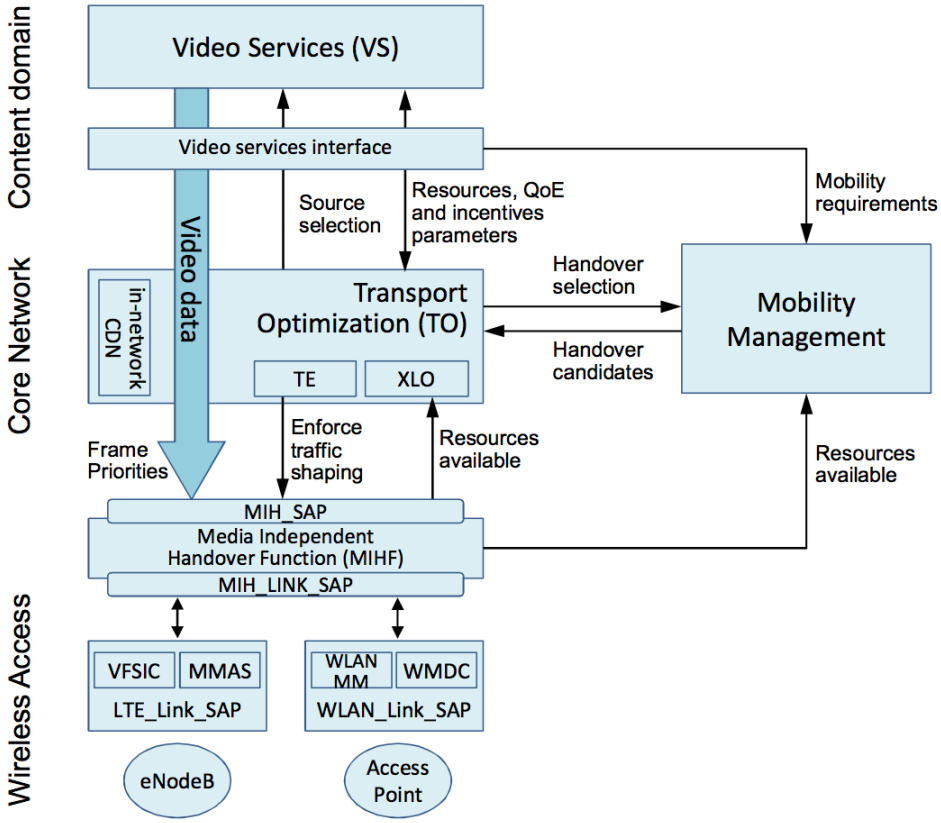

Figura 2.2: Modelo de referencia MEDIEVAL. Fuente [FP7]

por el WLANMM (Monitoring Module) y el WMDC (Dynamic Configuration).

El TO es un componente central de la arquitectura, mejorando el transporte desde el VS al usuario, utilizando las funcionalidades dentro de la red CDN y optimizando la cadena de entrega de vídeo en términos de recursos y la percepción de calidad de vídeo. De este modo, la optimización *cross-layer* de tráfico (XLO) coordina el reenvío del tráfico de vídeo a lo largo del camino de red y se encarga de las congestiones. En este último caso, el objetivo no se limita a los parámetros de QoS, pero logra una QoE global óptima para todos los usuarios afectados por la congestión.

El tráfico de video se puede adaptar a diferentes niveles de la ruta de vídeo: adaptación de contenido en la fuente (por ejemplo, mediante la selección de una codificación diferente de la de vídeo) o en la red (por ejemplo, descartar paquetes en el módulo de Traffic Engineering (TE)), o disminuyendo los recursos utilizados en el acceso inalámbrico y en la red central. Sin embargo, la información sobre la congestión se debe transmitir primero al TE. Por lo tanto, a pesar de que está perfectamente adaptado para hacer frente a congestiones en la red o al acceso inalámbrico, no es capaz de reaccionar a las variaciones dinámicas de las condiciones del canal rápidamente. Dicha adaptación tráfico gradual se realiza en el acceso inalámbrico por la VFSIC / WMDC que son conscientes de las condiciones actuales del canal. Esta operación se lleva a cabo después de que los paquetes han sido desencapsulado desde el túnel de GTP-U y antes de que se encapsulado en el protocolo PDCP.

Las diferentes capas de la arquitectura se coordinan entre sí para reaccionar a diferentes condiciones de tráfico. El TO recibe las notificaciones de diferentes eventos sobre las

condiciones actuales del enlace a través de la señalización MIHF. En una primera fase, el umbral del nivel de señalización del enlace está configurado, donde, después de la detección de malas condiciones del enlace, el enlace comienza a tirar paquetes, en base a la prioridad marcada en estos. En una segunda fase, un segundo umbral genera otra notificación del enlace al TO sobre el degradamiento de las condiciones de red y llamando a la gestión del tráfico basados en QoE de vídeo.

2.4. Interfaz abstracto Layer 2.5

Otro concepto introducido por el proyecto MEDIEVAL se conoce como interfaz abstracto Layer 2.5. Tiene un doble objetivo. Por un lado, sirve para encapsular las diferentes tecnologías de acceso inalámbrico que se espera que coexistan en los sistemas de comunicaciones del futuro. Esto se realiza por informar a las capas superiores, a las aplicaciones de vídeo en particular, de las funcionalidades disponibles en la capa inalámbrica de forma general. Al mismo tiempo, la información sobre la calidad del canal puede ser transferida a las capas intermedias (por ejemplo, componentes de movilidad y transporte).

Por otro lado, esta abstracción también proporcionará un interfaz para las aplicaciones de vídeo para configurar de manera óptima el comportamiento de la tecnología inalámbrica sobre la que operan. Por ejemplo, en el caso de las tecnologías inalámbricas basadas en reserva del medio, esta información se proporciona a las capas superiores para que puedan reservar los recursos necesarios de las aplicaciones de vídeo. En el caso de tecnologías que no admitan reservas pero permiten prioridades de flujo, las aplicaciones de vídeo pueden identificar las prioridades relativas de sus paquetes para aprovechar esta funcionalidad. Por lo tanto, el estándar Layer 2.5 permitirá que dichas funcionalidades de *cross-layer* entre las capas de red inalámbrica y de vídeo sin necesidad de conocer todos los detalles de cada uno las tecnologías inalámbricas en las que se basan, sólo las características relevantes para las interacciones con los servicios de vídeo.

Como resultado, la interfaz permitirá optimizaciones *cross-layer* mediante el intercambio de primitivas en ambas direcciones. Desde las capas superiores a las tecnologías inalámbricas, se proporciona las prioridades o los parámetros de reserva para la tecnología de acceso inalámbrico. Desde las tecnologías inalámbricas a las capas superiores, se informa de las capacidades de acceso inalámbrico y de disponibilidad, incluyendo también la movilidad y optimización del tráfico.

Esta idea comparte algunas similitudes con otras contribuciones (también hechas en algunos proyectos europeos), pero esta es la primera vez que se aplica al vídeo. Una de las novedades de este interfaz Layer 2.5, en comparación con otras propuestas, es que MEDIEVAL no tiene como objetivo proporcionar la misma abstracción entre las diferentes tecnologías. El objetivo es proporcionar un conjunto de interfaces abstractas que permitan a cada tecnología informar a las capas superiores sobre sus propias capacidades, y a su vez permita que las aplicaciones de video exploten estas capacidades mediante el uso de esta interfaz abstracta.

2.5. Gestión del tráfico de vídeo basado en la QoE

La gestión del tráfico en la red principal está diseñada para lograr una calidad óptima de la experiencia en todos los usuarios dadas las limitaciones de recursos de red. La red central tiene una visión general de las condiciones de la red y de tráfico actuales. El tráfico se gestiona dentro de los recursos de red disponibles antes de llegar al acceso inalámbrico. La gestión del tráfico se divide en dos funciones acopladas: la optimización basada en la calidad de la experiencia (en el XLO) y de conformación de tráfico (en el TE). La optimización basada en QoE es la inteligencia detrás de las acciones de adaptación de velocidad, impuesta por la modulación del tráfico. Para la adaptación de SVC en el *stream* de vídeo, la optimización indica qué capas en el destino que se deben tirar por el modulador de tráfico.

La optimización tiene como objetivo maximizar la QoE en general de varios usuarios mediante la asignación de tasa de bits para la transmisión óptima de los usuarios. Se consideran las condiciones del ancho de banda para estimar la tasa de bits máxima alcanzable por cada usuario. Para comprender el efecto de la gestión del tráfico en la QoE, las funciones de utilidad se emplean para describir la relación entre la tasa de bits de transmisión y calidad percibida en términos de valores de MOS (Fig. 2).

Para cada tasa de bits, se lleva a cabo una adaptación de vídeo y su correspondiente calidad de vídeo (QoE) son estimados mediante una medida de la calidad. En este trabajo, los vídeos están adaptados para tirar las capas de los vídeo codificados en SVC. La correspondiente QoE se estima por medio de la Video Quality Metric (VQM) [PW04]. La VQM tiene en cuenta los parámetros de las distorsiones en las dimensiones espaciales y temporales y se adopta en las Recomendaciones de la UIT [15]. La VQM da una estimación de la calidad percibida en términos de la Differential Mean Opinion Score (DMOS), que van desde 1 (sin correlación) a 0 (misma calidad), que describe la diferencia de percepción de calidad entre los vídeos originales y los vídeos procesados. Se asigna una Mean Opinion Score (MOS), que van de 1 (bajo) a 5 (alto) que describen el nivel de percepción absoluta: MOS = 5 - 4·DMOS.

El vídeo de esta prueba está codificado en 3 capas Coarse Grained Scalability (CGS) y 5 capas temporales. Cada punto de la curva contiene un conjunto de información: la combinación de capas, la tasa de bits de la combinación, y el valor MOS resultante. En las tres curvas de trazos de izquierda a derecha, el número de capas de CGS aumenta de 1 a 3 capas. En los cinco puntos de abajo hacia arriba en cada curva de trazos el número de capas temporales aumenta 1-5 capas. La función de utilidad es la envolvente que se muestra con la curva de puntos. En algunas velocidades de bit, existen múltiples combinaciones de capas, pero los valores MOS difieren en gran medida. Las combinaciones de las diferentes capas dan la calidad óptima de la experiencia.

Las funciones de utilidad son de contenido dependiente. En la figura 2.3 el vídeo muestra diferentes combinaciones de las capas y los diferentes requisitos de tasas de bits para lograr el mismo nivel de calidad de la experiencia. Sobre la base de las funciones de utilidad, la optimización basada en QoE asigna los diferentes *bitrates* de transmisión de vídeo para obtener un promedio de QoE óptima dentro del ancho de banda total disponible. Así, durante una congestión, el optimizador es capaz de evitar las capas más altas del vídeos puesto que son mas exigentes y ocupan demasiados recursos; manteniendo así un promedio de QoE óptima con el fin de mejorar la satisfacción de la mayoría de los usuarios.

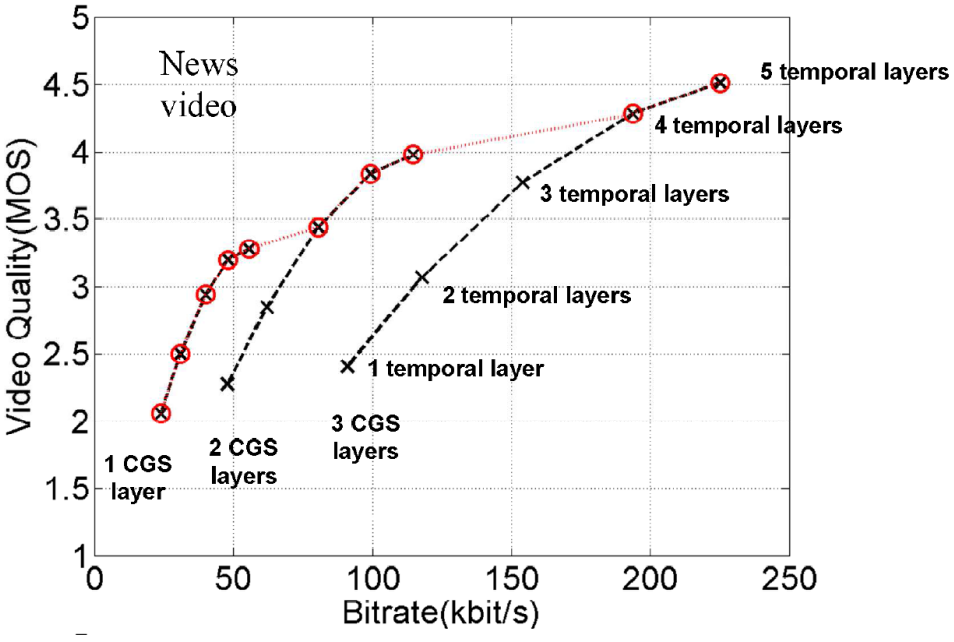

Figura 2.3: MOS sobre un vídeo codificado en SVC

Parte III

Descripción del trabajo realizado

Capítulo 3

Arquitectura del prototipo

3.1. Introducción

En este capítulo se va a describir la arquitectura empleada para mostrar los diferentes escenarios de la demostración. Así mismo, se describen cada uno de los elementos con sus respectivas funcionalidades.

3.2. Objetivos

La arquitectura propuesta para la demostración tiene como objetivos:

- Mostrar la adaptación de *bitrate* de vídeo en función de las condiciones de red sobre IEEE 802.11.
- Movilidad entre dominios de red que soporten o no optimización de vídeo.
- Visualizar qué sucede cuando no se tiene adaptación de *bitrate* de vídeo.

Dentro de estos objetivos, definiremos "dominio MEDIEVAL" al dominio de red que soporta adaptación y optimización del *bitrate* de vídeo; y definiremos "dominio no MEDIEVAL" al dominio que no tiene soporte para esta tecnología.

3.3. Elementos de la topología

En la topología propuesta podemos diferenciar tres tipos de elementos:

- Servidor de vídeo: está a la espera de los usuarios a que inicien la reproducción multimedia del contenido alojado. En esta demostración, el contenido consiste en un vídeo codificado en SVC llamado Big Buck Bunny.
- Mobile Access Router: ofrece acceso inalámbrico a los nodos móviles. Estos routers están conectados al servidor de vídeo y proveerán acceso a los nodos móviles a la red de cada operador. En el caso del router que da soporte a MEDIEVAL, se encargará de la optimización del tráfico de vídeo.

- Mobile Node: son los diferentes clientes que se conectan a la red inalámbrica provista por los MAR. Esto se pueden mover entre los dominios MEDIEVAL y no-MEDIEVAL dependiendo del escenario a recrear en la demostración.

Estos equipos se interconectan tal y como aparecen en la figura 3.1.

El objetivo de este prototipo es visualizar y recrear los diferentes escenarios donde un nodo móvil se conectará a la red con soporte para MEDIEVAL y visualizará el vídeo alojado en el servidor de multimedia Server2. Los diferentes nodos móviles se conectarán a la red de dominio MEDIEVAL y al no-MEDIEVAL para generar tráfico dentro de cada red y, con ello, una congestión para ver como varía la QoE en función del dominio en el que MN2 se encuentre reproduciendo el contenido.

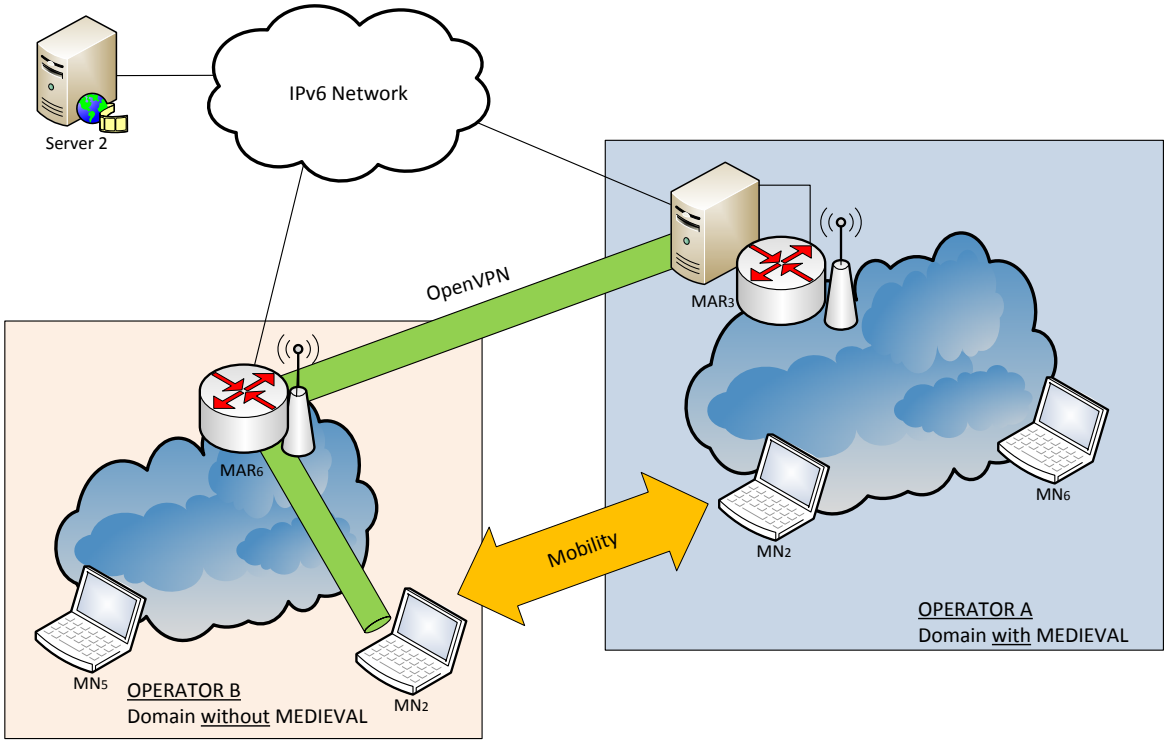

Figura 3.1: Topología de red del testbed

3.4. Plataforma de pruebas

El testbed se compone de seis equipos: cuatro PCs de sobremesa y dos PCs portátiles. A continuación se describe la funcionalidad de cada elemento y el equipo empleado en nuestro banco de pruebas.

3.4.1. Servidor de vídeo

El servidor de vídeo es el proveedor de servicio de multimedia. Es el elemento más simple de nuestro testbed y el que menos configuración necesita. Tiene como ruta por

defecto el núcleo de nuestra red IPv6 y monta un túnel IPv6-IPv4 debido a limitaciones de software por parte del servidor de vídeo. El problema está más descrito en la sección 4.2.4. En nuestra topología recibió el nombre de Server2.

3.4.2. Routers de acceso móviles

Son los elementos principales de nuestro banco de pruebas, en especial MAR3 que es el encargado de la optimización de vídeo y la movilidad de los nodos móviles entre el dominio MEDIEVAL y no-MEDIEVAL. Ambos routers van equipados con tarjetas de red inalámbricas Atheros 802.11 a/b/g para funcionar como punto de acceso. MAR3 incluye dos tarjetas de red: una para proveer acceso inalámbrico a los nodos móviles y la otra donde AirTime escuchará y medirá el medio; así mismo, MAR6 solo necesita una para funcionar como punto de acceso inalámbrico para el dominio no-MEDIEVAL. En nuestra demostración existían dos MAR:

- MAR3
- MAR6

En ambos casos, se emplearon equipos idénticos a Server2 con la misma configuración.

3.4.3. Nodos móviles

Estos equipos son los encargados de reproducir el contenido alojado en el servidor de vídeo (MN2) y generar tráfico hasta congestionar la red (MN5 y 6). MN2 será el dispositivo donde se va a medir la QoE del usuario puesto que se realiza la visualización del contenido tanto en el dominio MEDIEVAL como en el no-MEDIEVAL. Para la movilidad, MN2 contará con dos tarjetas de red inalámbricas mientras que MN5 y 7 solo cuentan con una. Para nuestra topología se emplearon tres nodos móviles:

- MN2
- MN5
- MN6

En el caso MN2, se empleó de nuevo una configuración de hardware igual al de Server2, MAR3 y MAR6.

Capítulo 4

Despligue del prototipo

4.1. Introducción

En esta sección se introducirán las herramientas empleadas para esta demostración ya existentes así como las utilidades desarrolladas específicamente para este prototipo con la finalidad de ser mostrado en la presentación.

En la tabla 4.1, se muestra los diferentes elementos de nuestra topología junto a los módulos instalados en cada uno de ellos.

Dispositivo	**Módulos**
Server2	- Librerías Live555 - Túnel IPv6-IPv4
MAR3	- Hostapd - Tcpdump - ODTONE - XLO - Flow Manager - AirTime - Servidor OpenVPN - Cliente Iperf
MAR6	- Hostapd - Tcpdump
MN2	- Librerías Live555 - Reproductor MPlayer - Cliente OpenVPN - ODTONE - Connection Manager - Tcpdump
MN5	- Librerías Live555 - Reproductor MPlayer - Cliente Iperf - Tcpdump
MN6	- Cliente Iperf

Tabla 4.1: Resumen de los módulos por cada elemento de la topología

4.2. Software empleado

4.2.1. Punto de acceso con Hostapd

Para hacer funcionar nuestro equipo como punto de acceso inalámbrico sobre el estándar IEEE 802.11g, empleamos el software `Hostapd`. Esta aplicación es un demonio que corre en segundo plano y funciona sobre un interfaz de red inalámbrico que nosotros elegimos a la hora de configurar los parámetros que ofrece proveyendo así acceso a la red.

4.2.2. Visualizador de tráfico con Tcpdump y Wireshark

`Tcpdump` es una herramienta que funciona bajo línea de comandos cuya utilidad principal es analizar el tráfico que circula por la red. Permite al usuario capturar y mostrar en tiempo real los paquetes transmitidos y recibidos de la red. Con esta utilidad podremos visualizar los diferentes interfaces de red para ver por donde recibe y encamina el tráfico. Nos será muy útil al momento de comprobar y demostrar que los traspasos en las pruebas de movilidad se han efectuado correctamente.

`Wireshark` es otra herramienta para visualizar el tráfico pero este cuenta con una GUI más cómoda de manejar a la hora de interpretar las capturas de tráfico de red.

La metodología consistió en realizar capturas en los diferentes interfaces de red con `tcpdump` para más tarde visualizar dichas capturas con `Wireshark`.

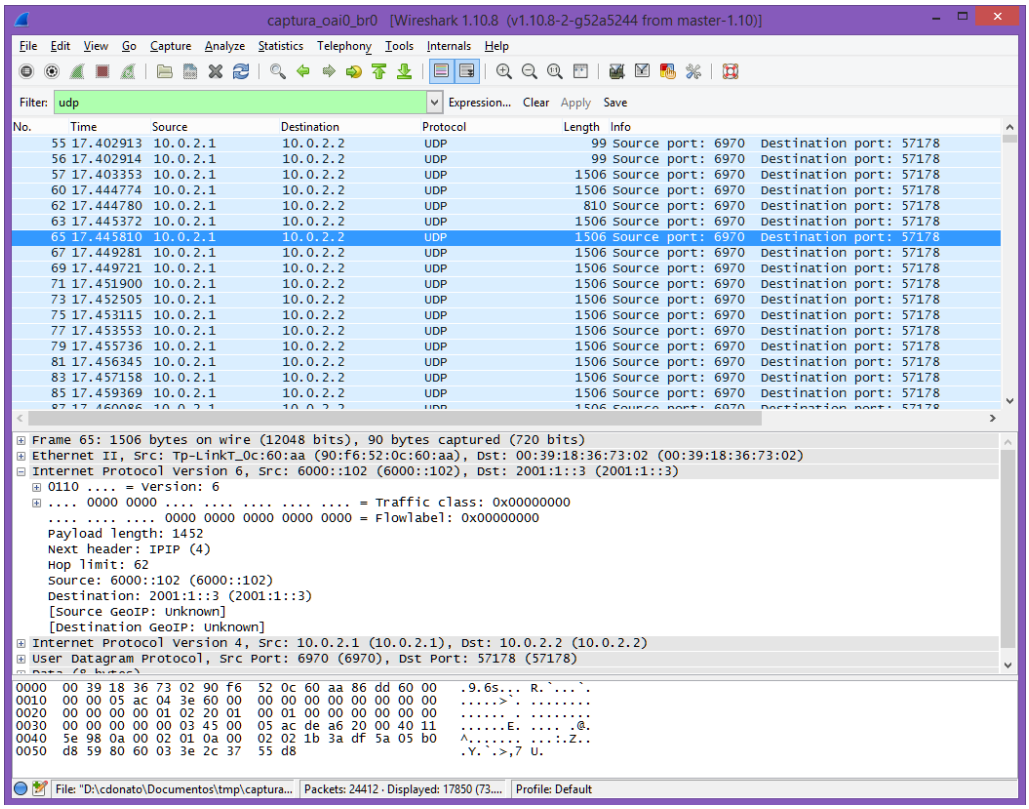

Figura 4.1: Captura de Wireshark mostrando el Túnel IPv6-IPv4

En la figura 4.1 podemos ver a Wireshark visualizando una captura realizada con

`tcpdump`.

4.2.3. Congestión de la red con Iperf

Uno de los objetivos de nuestra demostración es recrear el escenario donde varios usuarios estén demandando mucho tráfico para generar una congestión que afecte a nuestro nodo móvil donde se está visualizando el contenido desde el servidor de vídeo. Con tal fin, vamos a simular la congestión con un test de rendimiento de la red desde los MAR a un nodo móvil con `Iperf`, una herramienta destinada a tal fin. Con esta prueba de rendimiento, podremos ir variando el ancho de banda ocupado en la red por el nodo móvil hasta ver en qué punto empieza a afectar a la reproducción multimedia del contenido.

4.2.4. Túnel IPv6-IPv4

Este túnel nace como solución a las limitaciones del servidor multimedia Live555, empleado para ejercer de servidor de vídeo, ya que no soporta IPv6 y toda nuestra topología se basa en dicho protocolo. Para solventar dicho problema, encapsulamos el protocolo que originariamente está soportado, IPv4, en IPv6 con un túnel haciendo ver a la aplicación que funciona sobre IPv4 eliminando así dicha limitación.

Cada dispositivo que se conecte al servidor para comenzar la reproducción multimedia, deberá crear el otro extremo del túnel IPv6-IPv4 como es el caso de MN2.

4.2.5. OpenVPN

Este módulo provee el servicio multimedia cuando nuestro nodo móvil cambia a un operador que no soporta MEDIEVAL. Se consigue reencaminando el tráfico a través de una red privada a nuestro nodo móvil donde se está visualizando el contendo. De esta manera, se crea un túnel donde la dirección IPv6 sigue siendo la misma en el extremo del nodo móvil. Así sigue siendo alcanzable por el servidor de vídeo aunque estemos en otro dominio donde el rango de direcciones sea distinta variando únicamente las rutas entre los elementos intermedios haciendo el proceso transparente para nuestro servidor de contenido.

4.3. Software modificado

En esta sección se detallan los diferentes módulos que fueron modificados para las demostraciones de MEDIEVAL. La mayoría de los cambios fueron efectuados por los partners del proyecto o en otras demostraciones.

4.3.1. Librerías Live555 como transporte y servidor de vídeo

Las librerías de código abierto Live555 fueron usadas para la retransmisión de contenido multimedia e incluyen un servidor multimedia que hace uso de los protocolos RTP/RTCP/RTSP/SIP para la transmisión de dicho contenido.

Este servidor será el que empleemos para tal fin en nuestro *testbed* aunque tiene la limitación de no soportar el protocolo de red más reciente: IPv6. Esto es un inconveniente puesto que toda nuestra topología de red se basa en dicho estándar. En la sección 4.2.4 se explica la solución que incluimos para solventar este problema.

Se hicieron varias modificaciones para adaptar la MTU a la de nuestro túnel debido a que, por defecto, usaba 1500 bytes. Se añadió soporte para tomarla por parte del sistema operativo y evitar así la fragmentación IP visible antes de la modifación.

4.3.2. Reproductor multimedia MPlayer

Los nodos móviles visualizarán el vídeo alojado en Server2 con el reproductor MPlayer. En la sección B.5.2 se explica cómo configurar y habilitar soporte para el códec de vídeo SVC. Este reproductor se apoyará en las librerías Live555 para recibir el tráfico de vídeo y empleará el códec SVC para decodificar el contenido recibido.

Gracias a esta herramienta, podremos medir la QoE del usuario puesto que se reflejará aquí la congestión generada por los otros nodos móviles.

4.3.3. Interfaz Layer2.5: ODTONE

En MEDIEVAL se concibe el concepto de Layer2.5, un interfaz entre los niveles de enlace y de red. Con este interfaz se pretende que haya una mayor interacción entre la capa física y las capas de red y transporte.

Esta capa se implementa con un *framework* llamado ODTONE (*Open Dot Twenty One*), un software de código abierto para el protocolo de movilidad IEEE 802.21. Este módulo gestiona el tráfico entre los enlaces y la movilidad entre ellos. A él, se conectarán los clientes Flow Manager y Connection Manager descritos en la siguientes secciones.

4.4. Software desarrollado

4.4.1. Optimización de tráfico de vídeo con XLO

Las siglas XLO vienen de *Cross-Layer Optimizer*. Sin este módulo, esta demostración de MEDIEVAL no tendría sentido puesto que es el responsable de la optimización del tráfico de vídeo. Sus funciones son:

- Detectar el flujo de vídeo dentro del tráfico del nodo móvil.
- Identificar las capas del vídeo codificado en SVC.
- Según las condiciones de red, reducir el *bitrate* decidiendo qué capas de vídeo tirar para optimizar el tráfico y aumentar la QoE del usuario.

Para ello, el XLO accede a la cola de paquetes del *kernel* gracias a un par de reglas de `ip6tables`. Con esto consigue que al recibir un paquete pase primero por nuestro optimizador de tráfico, determine qué clase está manejando y si corresponde al flujo de

vídeo que soporta, decida si descartar o no capas de vídeo en función de las condiciones de red. Por último, recursar el tráfico según sus rutas hasta el destino.

Con objeto de determinar las condiciones de red, XLO se apoya de la herramienta desarrollada para esta demostración llamada AirTime. La sección 4.4.2 está dedicada a dicha utilidad que explica cómo funciona.

4.4.2. Detección de congestión con AirTime

Una de las partes más complejas de este prototipo es la detección del exceso de tráfico que puede llegar a afectar la QoE del usuario al momento de la visualización del contenido. Con este fin se ideó AirTime, una aplicación que monitoriza y mide el tiempo de uso en el aire por las tramas de red inalámbricas.

Fue desarrollado en el lenguaje de programación C ya que se apoya en las librerías `libpcap` para acceder al tráfico de nivel de enlace del interfaz. Se valoraron otros lenguajes de programación y entornos pero nos decantamos por C y `libpcap` ya que existen utilidades como `tcpdump` que están escritas sobre la misma librería que AirTime y nos garantizan el funcionamiento y comportamiento esperado.

A la hora de escuchar una trama conocemos la velocidad a la que fue transmitida y conocemos la longitud de esta por lo que podemos deducir el tiempo que ha empleado dicha trama en ser transmitida y, como consecuencia, el tiempo que ha ocupado el medio. Para medir el tiempo de uso del canal nos basamos en la velocidad de transmisión de un paquete y despejamos el tiempo de transmisión:

$$R_b = \frac{L_{trama}}{T_{trans}} \Rightarrow T_{trans} = \frac{L_{trama}}{R_b}$$

Como los canales del estándar IEEE 802.11 no son privados, cualquier otra red que esté operando en el mismo canal nos va a interferir a la hora del envío y recepción de tramas, congestionando el medio e influyendo en nuestra transmisión de contenido de vídeo, por lo que la entrega de vídeo no está afectada únicamente por las congestiones introducidas en nuestra propia red sino por todos los dispositivos que estén usando el mismo canal al mismo tiempo.

Con estas premisas se diseñó e implementó AirTime. Para operar correctamente, es necesario configurar el interfaz de red inalámbrico sobre el que va correr en modo monitor. Dicho modo, nos permite escuchar un canal y saber quién, cuando y cómo está empleando el medio dentro. Al momento de acceder a dichos paquetes con `libpcap`, estos son subido a la aplicación con una cabecera añadida por el *kernel* llamada *radiotap*. Esta cabecera nos añade información sobre la capa física como la velocidad de transmisión, modulación empleada, canal usado, relación señal a ruido al momento de recibir, etc. En la figura 4.2 podemos ver los diferentes campos de esta cabecera desde una captura de `Wireshark`.

Una vez recogidos los datos, AirTime calcula el tiempo que se está usando el canal y reporta dichos valores al XLO con la frecuencia que nosotros le indiquemos. Para la demostración estimamos que 500 ms era el valor óptimo.

Este módulo fue desarrollado en este Trabajo de Fin de Grado por parte de la Universidad Carlos III de Madrid para solucionar la problemática de la detección de congestión en redes basadas en IEEE 802.11.

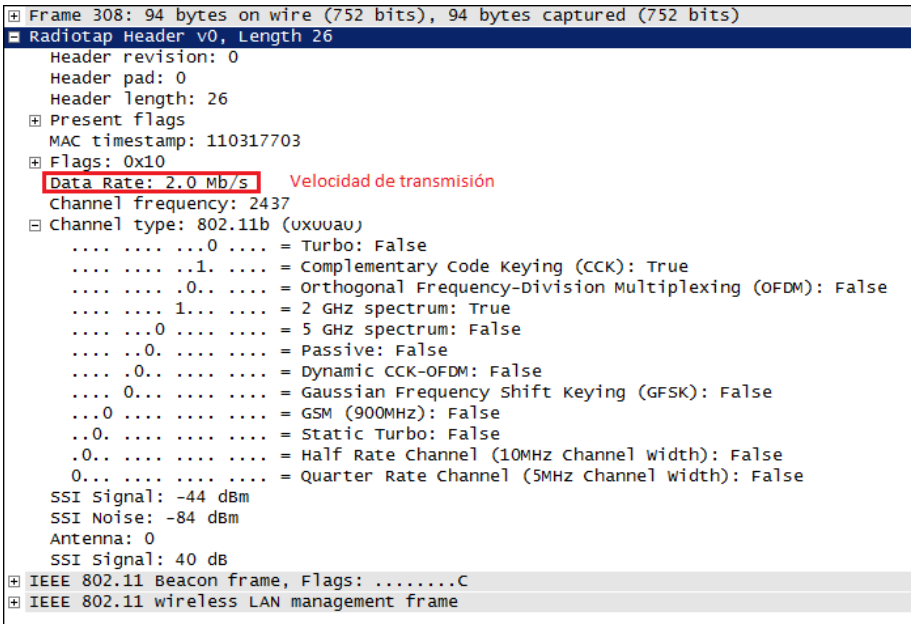

Figura 4.2: Captura de la cabecera *radiotap*

4.4.3. Movilidad entre operadores con Flow Manager

La movilidad entre operadores se gestiona a través del protocolo IEEE 802.21. Este protocolo está diseñado para dar soporte a la movilidad entre diferentes tecnologías de acceso a la red como IEEE 802.11, 3GPP o 3GPP2. Con este objetivo se desarrolló la herramienta Flow Manager, una utilidad que gestiona los *handover* (traspasos) entre dominios de red controlando los flujos de datos gestionados por MAR3 para que en el momento que un dispositivo necesite cambiar de dominio, sea capaz de reencaminar el tráfico actual a la nueva ubicación sin interrupción. Para el desarrollo, se empleó el *framework* ODTONE (*Open Dot Twenty ONE*) que es una implementación de código abierto del estándar IEEE 802.21.

La configuración de este módulo está detallada en el Apéndice C.3.5.

4.4.4. Movilidad con Connection Manager

La gestión de movilidad es llevada a cabo por Connection Manager, una herramienta basada en el *framework* ODTONE para el soporte del protocolo de movilidad IEEE 802.21 al igual que lo visto en la sección 4.4.3 de MAR3. Cuando MN2 detecta que se va a mover a otra red lo hace a través del Connection Manager que, a su vez, le indica al Flow Manager en MAR3 a donde se va trasladar. Se desarrolló una interfaz gráfica para la demostración que hacía mas intuitivo y visual el *handover* entre los dos dominios tal y como se puede ver en la figura 4.3.

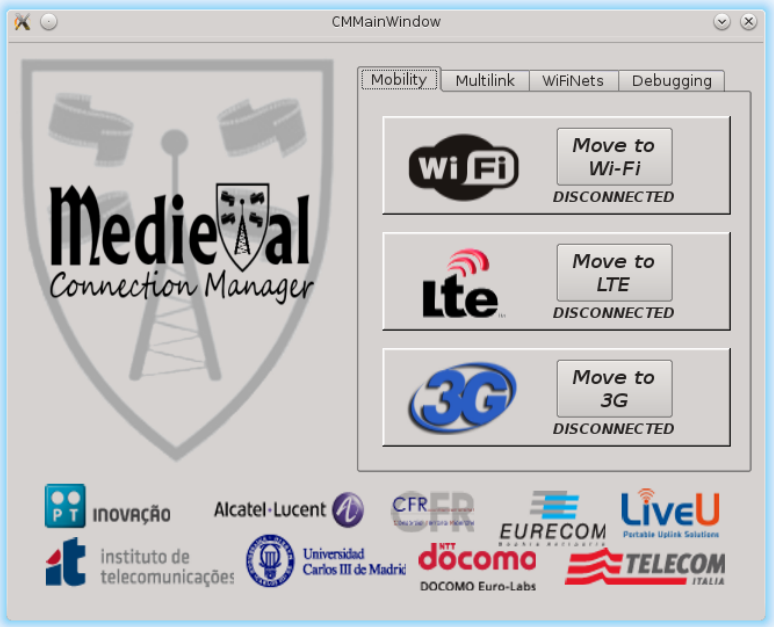

Figura 4.3: Captura de Connection Manager GUI en MN2

4.5. Pruebas de validación

Antes de recrear los escenarios para la demostración, se hicieron diferentes pruebas de validación para comprobar el funcionamiento esperado de cada módulo, así como la integración entre ellos y el ajuste de los parámetros necesarios.

4.5.1. Integración y validación de los módulos

El primer paso fue comprobar que la combinación túnel IPv6-IPv4, librerías Live555 para el servidor y transporte del vídeo junto al reproductor de medios MPlayer funcionaban correctamente. Para probarlos, se dispusieron un nodo móvil, un MAR proveyendo red inalámbrica y el servidor de vídeo. Configuramos el túnel, lanzamos el servidor de vídeo y comenzamos el streaming de vídeo en el nodo móvil.

Una vez comprobado que el nodo móvil era capaz de visualizar el contenido sin problemas, se procedió a hacer la prueba anterior pero esta vez el nodo móvil recibía el flujo de vídeo a través de la VPN, necesario para la movilidad entre operadores. El MAR hacía de servidor VPN y el nodo móvil hacía de cliente, conectándose y visualizando, de la misma manera que en la prueba anterior, el vídeo.

Estas dos pruebas nos aseguran que el streaming de vídeo funcionaba correctamente para el escenario donde el nodo móvil iba a sufrir congestiones en el dominio MEDIEVAL y en el escenario donde la optimización de tráfico de vídeo no estaba soportada.

El siguiente paso fue probar que las herramientas XLO y AirTime se comunicaban entre ellas correctamente y comprobar el descarte de paquetes por parte del XLO a partir de los datos recogidos por parte de AirTime. De nuevo, se dispuso el servidor de contenidos, un

MAR que proveía red inalámbrica a la vez que corría la combinación XLO y AirTime y de un nodo móvil que se conectaba al MAR para reproducir el contenido. Dicha combinación de herramientas necesita saber los umbrales para los que se detecta congestión pero, en este caso, queríamos que se dieran falsos positivos para probar que hacen bien su trabajo, por lo que se indicaron umbrales muy bajos de congestión para probar que el nodo móvil recibía el vídeo y calidad de imagen variaba.

En esta prueba concluimos que: AirTime medía correctamente el medio inalámbrico; que el XLO hacía descarte de paquetes correctamente y que, además, lo estaba efectuando en función de los datos recolectados por AirTime.

Con todas las pruebas anteriores superadas, la última fue probar la movilidad entre operadores con la combinación Flow Manager y Connection Manager. Se dispuso de dos equipos MAR, el servidor de vídeo y un nodo móvil. El primer equipo MAR corría el servidor VPN, el Flow Manager y proveía red inalámbrica mientras que el MAR restante, limitaba su función a proveer conectividad inalámbrica. Una vez todo iniciado, el nodo móvil reproducía el contenido del servidor de vídeo y se movía entre ambos dominios a través del Connection Manager.

Para validar esta prueba se capturó mediante `tcpdump` los interfaces de red inalámbricos de ambos MAR para probar que el tráfico se reencaminaba en función de donde el nodo móvil se encontrara. La figura 4.4 muestra la combinación de ambas capturas superpuestas.

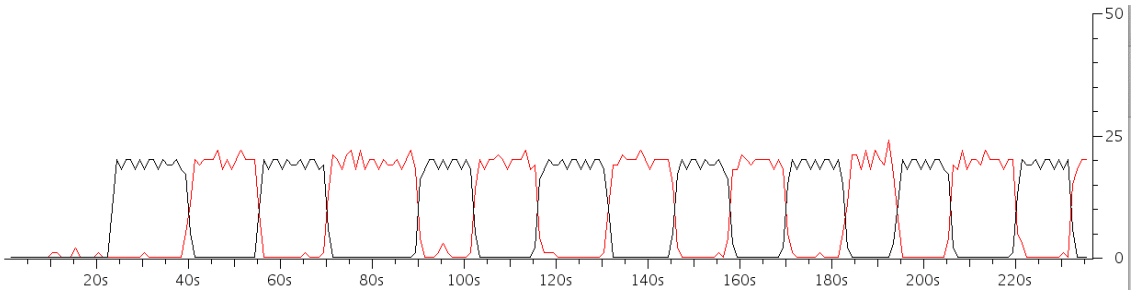

Figura 4.4: Captura de los *handovers* entre operadores

4.5.2. Ajuste de parámetros

Una vez comprobado que todas las herramientas estaban correctamente integradas y que el resultado inicial era el esperado, se procedió a ajustar los diferentes parámetros de dichas herramientas.

Para ello se recreó un escenario donde un nodo móvil se conectaba al dominio con soporte para MEDIEVAL y se visualizaba un vídeo en el nodo móvil desde el servidor de contenidos. Se lanzó el servidor de vídeo, arrancamos el XLO y AirTime en un MAR y lanzamos el túnel IPv6-IPv4 en el nodo móvil.

Lo primero fue visualizar el vídeo sin que hubiera ningún tipo de descarte de paquete mientras que hicimos una captura con `tcpdump` en la interfaz de red inalámbrica del nodo móvil que recibía el flujo. Con dicha captura, sacamos una gráfica para ver como variaba el *bitrate* del vídeo y así ajustar el ancho de banda máximo provisto por MAR.

Esta gráfica la podemos ver en la figura 4.5. Las unidades del eje ordenadas son bits mientras que las del eje de abscisas son segundos.

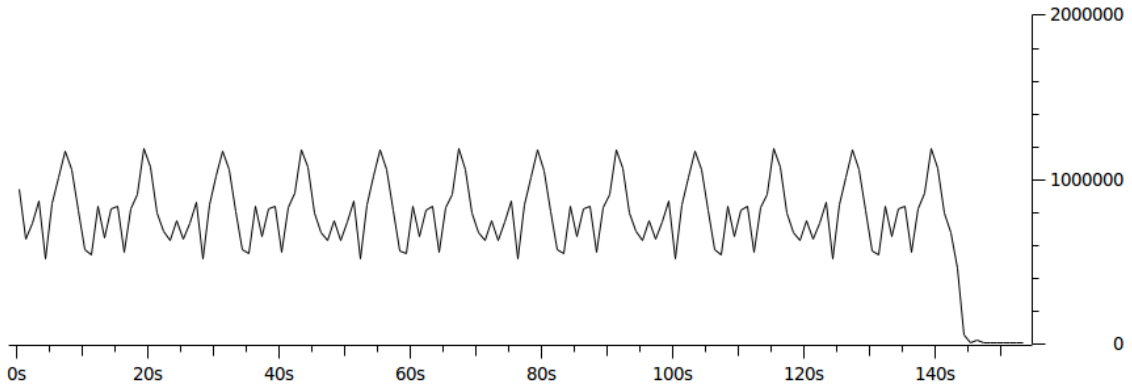

Figura 4.5: Gráfica del *bitrate* del vídeo empleado en la demostración

De gráfica sacamos los valores máximos de ancho de banda que podía tomar el vídeo preparado para nuestra demostración 1.2 Mbps y cuál era el *bitrate* medio de alrededor de 1 Mbps. Por esto, consideramos limitar el interfaz de red de los MAR a 2.7 Mbps con el comando `tc`, algo mas del doble que ocupa el pico máximo del *bitrate* de nuestro vídeo para tener ancho de banda suficiente para la congestión.

Para validar que realmente se limitaba el interfaz inalámbrico de los MAR a la velocidad que nosotros queríamos, se realizó un `Iperf` en modo TCP y comprobamos que la velocidad del test fuera la misma que la impuesta al interfaz.

Lo siguiente fue determinar los umbrales de congestión para la combinación AirTime junto al XLO de detección. Para determinarlo, se lanzó el vídeo en el nodo móvil y se anotó cuánto tiempo en el aire ocupaba cuando se trasmitía por cada prueba realizada de las veinte que realizamos. Tomadas las medidas, se procedió a congestionar progresivamente el medio con `Iperf` desde otro nodo móvil al mismo tiempo que se visualizaba el contenido y anotar para qué valores el vídeo comenzaba a degradarse.

Tras otras pruebas se tomaron los valores óptimos para los que XLO debía comenzar a descartar una capa o dos en función de la congestión. En la figura 4.6 se muestra a AirTime corriendo.

Por defecto, AirTime muestra todas las estaciones que están funcionando sobre el canal seleccionado (y los canales que se solapan), el número de paquetes capturados, el porcentaje de uso de cada una de las estaciones, el tiempo total del canal usado y el porcentaje total sobre el empleo del canal. Nosotros anotamos el tiempo total del canal usado cuando se visualizaba el vídeo con problemas mientras que se congestionaba la red.

```
Mac Address             pkt_send     pkt_rcv      pkt_tot       (%) total
FF:FF:FF:FF:FF:FF       0            43           43            4.246399
74:2F:68:72:A3:E9       1            0            1             0.000000
00:07:88:C4:6C:A6       3            0            3             0.000000
C4:17:FE:67:A0:BB       1            0            1             0.000000
20:16:D8:02:72:A6       1            0            1             0.000000
44:6D:57:3B:53:2F       2            0            2             0.000000
E0:CA:94:FA:D6:6F       1            0            1             0.000000
68:94:23:00:93:73       1            0            1             0.000000
70:F1:A1:05:A1:EF       1            0            1             0.000000
00:26:CB:C6:8B:F1       9            0            9             0.000000
FC:F8:AE:C0:AC:24       3            0            3             0.000000
3C:77:E6:E3:65:07       1            0            1             0.000000
70:1A:04:E9:BB:B3       1            0            1             0.000000
84:A6:C8:3D:8F:1F       2            0            2             0.000000
90:4C:E5:B3:11:F0       3            0            3             0.080533
60:67:20:2A:58:90       2            0            2             0.000000
4C:3C:16:2A:99:B6       4            0            4             0.000000
7C:C5:37:C2:04:2C       1            0            1             0.000000
0C:84:DC:05:2D:3E       2            0            2             0.000000
88:53:2E:6C:A8:81       1            0            1             0.000000
7C:E9:D3:37:48:6C       1            0            1             0.000000
00:08:CA:A0:A2:92       2            0            2             0.008000
AC:81:12:4F:74:B8       3            0            3             0.033622
74:E5:43:AD:2D:FE       1            0            1             0.068800
00:19:A9:2F:EC:00       1            0            1             0.022933
24:FD:52:6E:6A:5E       1            0            1             0.006100
8C:29:37:1D:C0:D1       1            0            1             0.019644
CC:08:E0:D0:BD:15       1            0            1             0.014756
18:9E:FC:37:65:0B       1            0            1             0.014756
E0:06:E6:29:1E:8A       1            0            1             0.006167
1C:E6:2B:EB:9D:3E       1            0            1             0.015600
F8:1E:DF:DF:32:70       2            0            2             0.025244
Total stations: 32
Total time used over 500 ms: 26.597658 ...  total: 5.319531
Time to report!
```

Figura 4.6: Captura de AirTime funcionando

Capítulo 5

Escenarios de prueba

5.1. Introducción

El objetivo de este Trabajo de Fin de Grado es el desarrollo de un prototipo que recrea diferentes escenarios en los que se simulan y se mide la QoE del usuario cuando un nodo móvil experimenta diferentes condiciones en la red inalámbrica. Por esto, los objetivos de este prototipo son:

- Evaluar y medir la QoE en un nodo móvil cuando reproduce un vídeo en el dominio MEDIEVAL mientras que otro usuario varía las condiciones de red introduciendo una congestión progresiva.
- Movilidad entre operadores con soporte de MEDIEVAL y sin él.
- Repetir las medidas de la QoE cuando el nodo móvil se mueve a un operador sin soporte para MEDIEVAL y se recrea el primer escenario.
- Evaluar y medir la QoE en dos nodos móviles simultáneamente sobre el operador con MEDIEVAL cuando reproducen el mismo contenido mientras que otro usuario inicia una nueva congestión.

5.2. Medición de la QoE en dominio MEDIEVAL

En esta primera prueba, el objetivo es evaluar y medir la QoE del usuario cuando el nodo móvil se encuentra reproduciendo contenido en una red con dominio MEDIEVAL. El escenario planteado es el de la figura 5.1.

El nodo móvil MN2 se conectará a la red provista por MAR3 y comenzará la reproducción de un vídeo codificado previamente con SVC alojado en Server2. En la figura 5.2a podemos visualizar la calidad inicial del vídeo sin la reducción de *bitrate* ni descarte de capas.

Al cabo de unos segundos de la reproducción del medio, comenzaremos una congestión progresiva donde el nodo móvil MN6 realizará una prueba del ancho de banda con `Iperf` en la que se aumentará poco a poco la velocidad máxima de la prueba de rendimiento. Con esto último, conseguiremos congestionar la red de manera progresiva hasta que el router

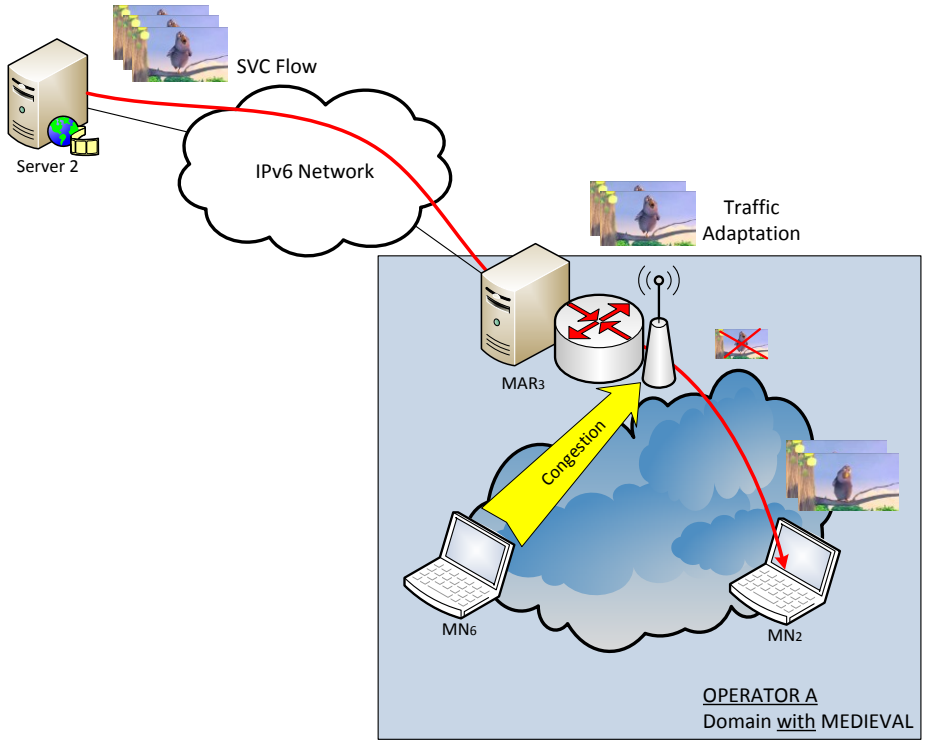

Figura 5.1: Escenario 1 de la demostración.

móvil MAR3 detecte que la congestión es suficiente como para que empiece a descartar una capa del vídeo. Este descarte de capa implica una reducción mínima en la calidad de imagen del vídeo pero reduciremos el ancho de banda evitando la degradación de la QoE del usuario o, incluso, interrupciones en la reproducción del medio. La figura muestra un *frame* del vídeo con dos capas.

Se mantendrá la congestión durante un tiempo y, cuando esta finalice, MAR3 detectará que el medio ha sido descongestionado retomando de nuevo la calidad inicial puesto que el ancho de banda disponible ahora es suficiente como para que MN2 pueda visualizar el vídeo a máxima calidad sin degradación.

Una vez probado que MAR3 es capaz de detectar congestión del medio y ajustar el vídeo en función del ancho banda, se procede a congestionar agresivamente el medio ocupando casi todo el canal. En esta situación, MAR3 identificará que apenas queda canal disponible por lo que procederá a descartar dos capas del vídeo resultando un descenso significativo de la calidad de imagen pero el usuario aún puede visualizar el vídeo fluidamente. La figura 5.2c es un *frame* del vídeo cuando sólo se reproduce una capa.

Como en el caso anterior, se mantendrá la congestión durante un tiempo y cuando finalice, MAR3 volverá a detectar que la congestión ha terminado y puede dejar de descartar capas del vídeo para recuperar la calidad original.

Al recrear este escenario, se capturó el tráfico recibido en el interfaz de red inalámbrico del nodo móvil MN2 con `tcpdump` para visualizar la reducción de *bitrate* según las capas de vídeo recibidas y compararlas con la gráfica original 4.5.

(a) Máxima calidad de imagen. (b) Reducción de una capa.

(c) Reducción de dos capas. (d) *Frame* deteriorado.

Figura 5.2: Frames de las diferentes calidades de vídeo

La figura 5.3 muestra la gráfica de cómo varía el *bitrate* en función de las capas descartadas por el XLO.

Si analizamos la gráfica detenidamente podemos diferenciar dos zonas. La primera es desde el segundo 18 cuando se comienza la reproducción del vídeo por parte del nodo móvil MN2 hasta el segundo 40. A partir del segundo 40 se aprecia un descenso del *bitrate* progresivo hasta el segundo 70 aproximadamente. En este intervalo se realizó una congestión desde el nodo móvil MN6 donde el XLO adaptó el *bitrate* para que el vídeo pudiera ser visualizado por el usuario según las condiciones del medio inalámbrico, gracias a las mediciones de AirTime. La congestión se detuvo y, por lo tanto, no existen condiciones desfavorable por lo que se devuelve el vídeo a su calidad máxima tal y como se ve desde el segundo 70 aproximadamente hasta segundo 87.

Se produjo de nuevo otra congestión pero esta vez mas progresiva ya que la adaptación del *bitrate* es menos pronunciada desde el segundo 120 hasta el 130 donde se recupera de nuevo la calidad original.

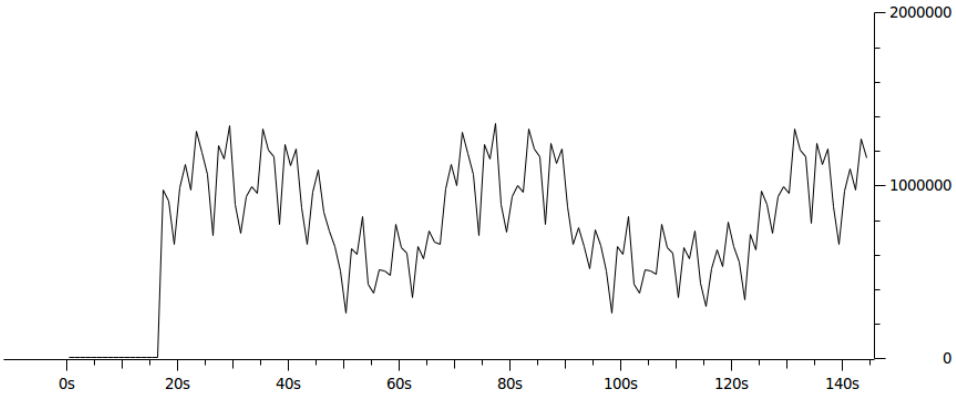

Figura 5.3: Gráfica del *bitrate* del vídeo con descarte de capas.

5.3. Movilidad entre operadores

La siguiente prueba tiene como objetivo demostrar la capacidad de MEDIEVAL de seguir proveyendo servicio aún cuando el operador de red no lo soporta. La figura 5.4 muestra el escenario que recreamos.

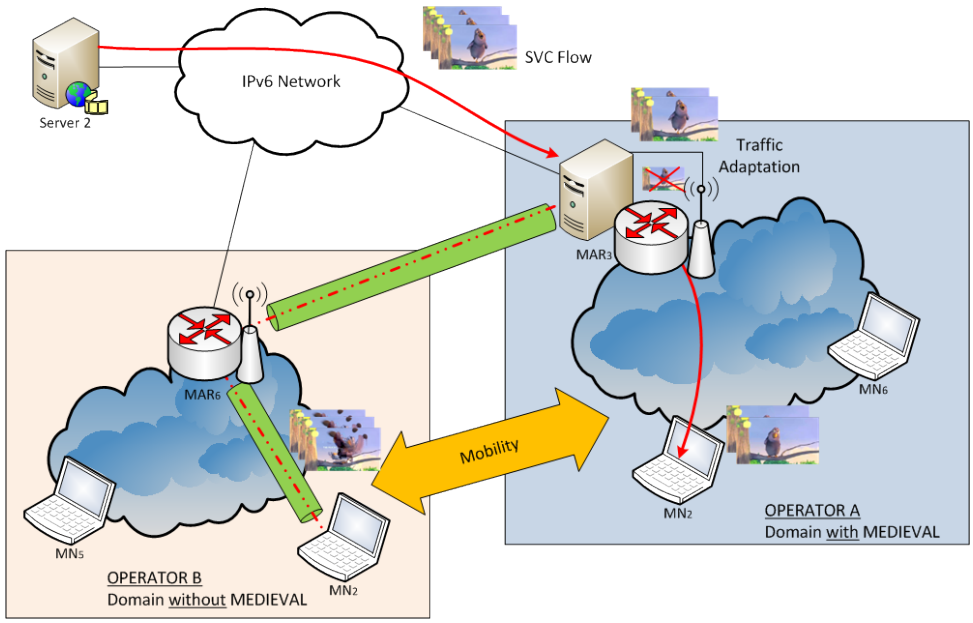

Figura 5.4: Escenario 2 de la demostración.

En esta prueba, el nodo móvil MN2 estará reproduciendo el vídeo desde Server2 en la red inalámbrica provista por MAR3 con dominio MEDIEVAL y, a través del Connection Manager, se moverá a la red de MAR6. Al momento de recibir el mensaje de *handover* en MAR3 por parte del nodo móvil, éste confirmará el traspaso y reencaminará el tráfico por la OpenVPN donde MN2 podrá seguir recibiendo el *streaming* de vídeo sin cortes.

Una vez confirmado que el *handover* se ha realizado correctamente y se ha visualizado el tráfico en el interfaz de red creado por OpenVPN, se procede a volver al dominio original y mostrar como el flujo de vídeo vuelve a ser gestionado por MEDIEVAL. Adicionalmente, se realizaron cuatro *handovers* para probar que la movilidad funcionaba sin interrupciones.

La figura 4.4 muestra una gráfica de cómo el tráfico se mueve entre los interfaces de red según el dominio al que estemos conectados.

5.4. Medición de la QoE en dominio no-MEDIEVAL

La finalidad de esta prueba es mostrar como sería la QoE de un usuario en una red convencional donde MEDIEVAL no introduce ninguna mejora. En la figura 5.5 se enseña la configuración empleada para recrear este escenario.

Nuestro MN2 se moverá a la red de MAR6 como se ha ejemplificado en la prueba anterior. Como en la primera prueba 5.2, MN2 visualizará el vídeo desde Server2 durante toda la prueba. Esta reproducción del contenido será siempre a máxima calidad durante todo el proceso puesto que MAR6 no soporta optimización del flujo de vídeo como MAR3

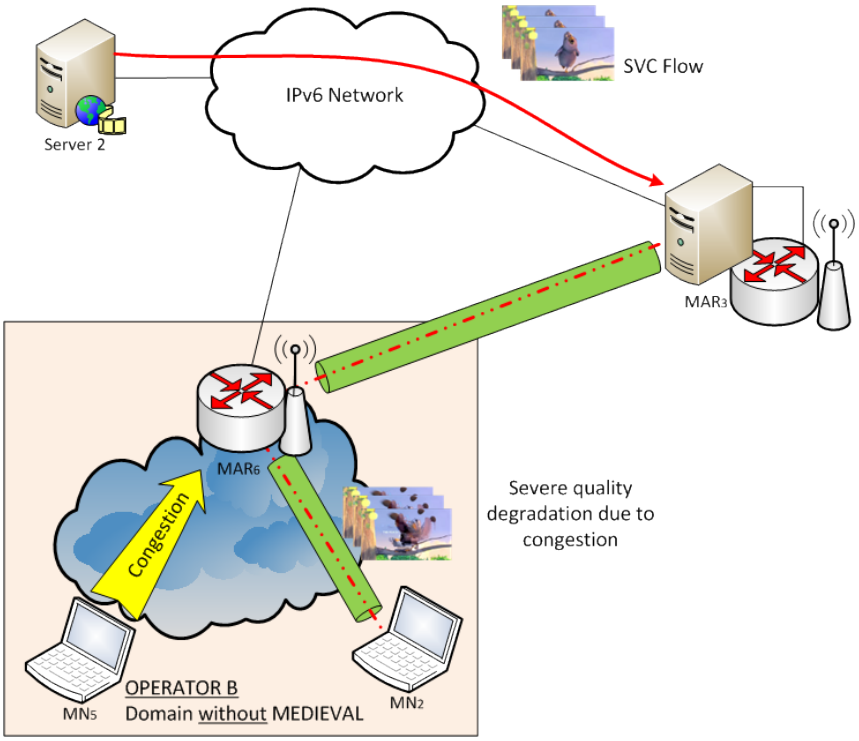

Figura 5.5: Escenario 3 de la demostración.

con la solución MEDIEVAL y, por lo tanto, pero será perceptible a la congestión.

Para probarlo, MN5 comenzará a congestionar el medio inalámbrico con `Iperf` progresivamente hasta que la calidad del vídeo empiece a degradarse junto a la QoE se vea deteriorada debido a la interferencias introducidas y, puesto que la adaptación del *streaming* no está habilitada, en un momento determinado llegue incluso a pararse.

Después de varias pruebas en este escenario, el vídeo se pixelaba, los *frames* se visualizaban desordenados o, en el peor de los casos, no era capaz de seguir la reproducción parando la conexión.

En la figura 5.2d se muestra un ejemplo de un *frame* deteriorado debido a la congestión introducida por MN6.

Con este escenario se muestra cómo la QoE decae de manera que llega a ser imposible visualizar el contenido dónde MEDIEVAL es capaz de entregar dicho contenido.

Al igual que en el primer escenario, se capturó con `tcpdump` el tráfico en el interfaz inalámbrico del nodo móvil MN2. La figura 5.6 muestra la gráfica de cómo varía el *bitrate* del vídeo en recepción llega hasta cero en determinados puntos de la reproducción coincidiendo con los momentos que se hace casi imposible la reproducción del medio.

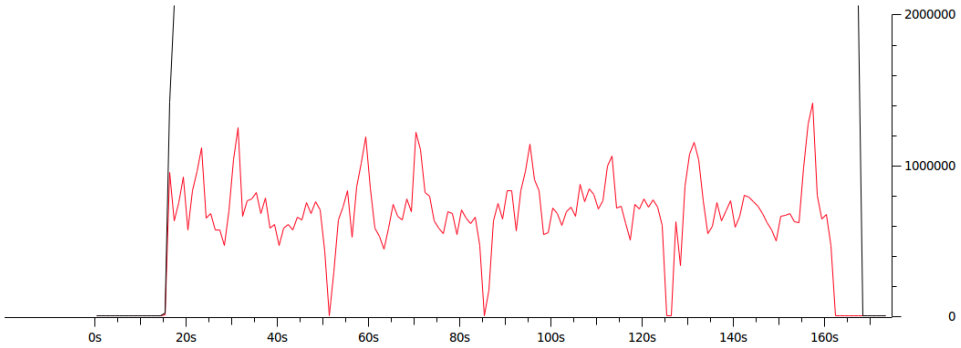

Figura 5.6: Gráfica del *bitrate* del vídeo con congestión.

5.5. Medición de la QoE con varios usuarios

Para terminar la demostración, preparamos un último escenario en la que dos nodos móviles, MN2 y MN6, reproducen el mismo contenido a la vez. La figura 5.7 muestra la topología empleada para esta prueba.

Los nodos móviles comenzarán la reproducción del mismo vídeo mientras que MN5 introducirá una congestión progresiva para mostrar el comportamiento del XLO en esta situación. El XLO irá adaptando cada flujo de vídeo en función de la demanda total por ambos usuarios más la congestión introducida.

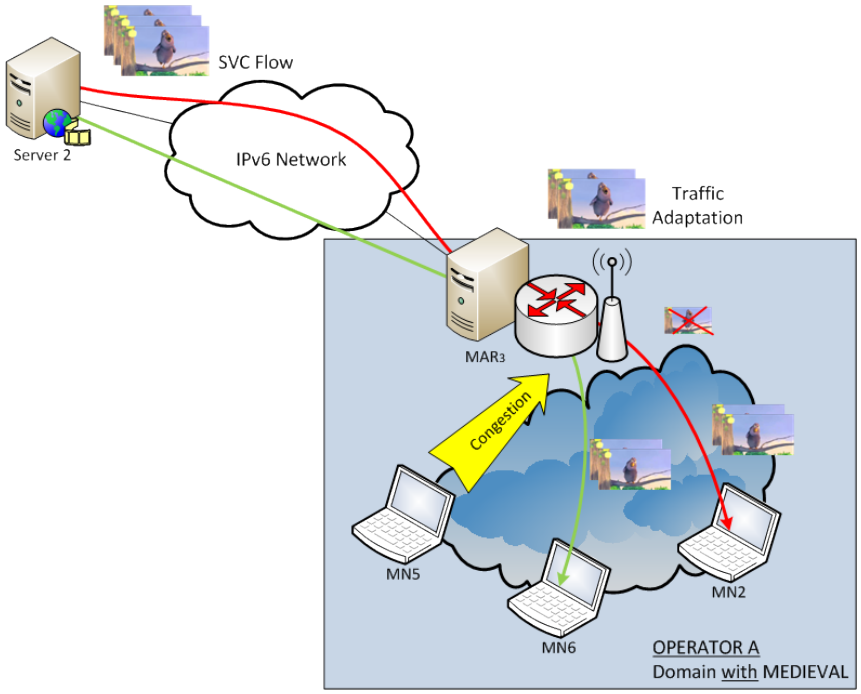

Figura 5.7: Escenario 4 de la demostración.

Parte IV

Conclusiones y trabajos futuros

Capítulo 6

Conclusiones y trabajos futuros

6.1. Conclusiones

El objetivo de este Trabajo de Fin de Grado ha sido el desarrollo y montaje de una solución para la optimización del tráfico de vídeo en redes inalámbricas. Esta demostración formó parte del proyecto europeo MEDIEVAL y fue mostrada en la auditoría final de Sophia Antipolis el pasado 11 de Octubre de 2013.

El desarrollo comenzó con el análisis del diseño propuesto por MEDIEVAL. Se enfocó en mostrar los posibles casos a los que se podía enfrentar un usuario a la hora de reproducir contenido de vídeo y cómo esta solución solventaba dichas situaciones. Por ello, se planificó la presentación en cuatro escenarios.

Una vez estuvo decidido qué se quería mostrar, se propuso la topología y los elementos que la iban a componer para más tarde decidir el software disponible que podíamos usar y qué necesitábamos desarrollar.

El siguiente paso fue montar y configurar el testbed según la topología propuesta. Después, se integró en la solución los módulos ya desarrollados y se verificó el correcto funcionamiento de éstos.

Probada la integración entre las herramientas existentes, se comenzó el desarrollo del resto de utilidades necesarias. De este modo, se desarrollaron por parte de los partners del proyecto las herramientas Flow Manager, Connection Manager y el XLO dejando pendiente la problemática de la detección de congestión a la Universidad Carlos III de Madrid. Desde aquí, se diseñó y desarrolló AirTime como mecanismo de detección de congestión en redes inalámbricas basadas en IEEE 802.11, el cual se combinaría con el XLO para la optimización del tráfico de vídeo en tiempo real.

Teniendo todas las herramientas ya integradas en nuestro testbed, se pasó a realizar pruebas de validación para comprobar que la cadena completa de módulos funcionaba según lo esperado. Con todos los módulos desplegados, se ajustaron los umbrales de detección en AirTime midiendo cuándo el vídeo comenzaba a mostrar problemas en la reproducción del medio según el nivel de congestión para que el XLO entrase a operar.

Con todo listo, se realizaron una batería de pruebas con el fin de sacar diversas gráficas donde se mostraba como el *bitrate* del vídeo se variaba en función de las condiciones de red inalámbrica y, además, se midió la QoE en el dominio donde MEDIEVAL estaba operativo

comparándolo con el escenario donde no era soportado.

Con los objetivos cumplidos, se migró toda la configuración del banco de pruebas de Eurecom, afincado en Sophia Antipolis, lugar donde tuvo la presentación. Allí se volvieron a validar todas la pruebas realizadas en el banco de pruebas de la UC3M y se compararon los resultados obtenidos.

Para finalizar, la demostración fue mostrada a los revisores de la Unión Europea que quedaron satisfechos con el esfuerzo puesto por parte de todos los integrantes del proyecto, remarcando el gran trabajo realizado en las demostraciones y calificando el proyecto como "Excelente" [OWS] .

Se concluye, por tanto, que los objetivos iniciales propuestos para este Trabajo de Fin de Grado se cumplieron.

Figura 6.1: Fotografía de la presentación en Sophia Antipolis.

6.2. Trabajos futuros

En la siguiente lista, se proponen varios trabajo a implementar en un futuro:

- Implementación e integración real por parte de los operadores en sus infraestructuras de red.
- Uso de diferentes tecnologías de acceso a la red como, por ejemplo, UMTS, LTE, LTE-A, WiMax...

- Creación de diferentes servicios multimedia enfocados a la arquitectura propuesta:
  - Servicios de videoconferencia con codificación en tiempo real de vídeo en SVC para delegar en la red la gestión óptima del tráfico de vídeo y lo adapte en función a las condiciones de los interlocutores.
  - Servicios de vídeo donde se comience a reproducir el contenido en una red y, al moverte a otra, la red sea capaz de seguir proporcionado servicio sin que el usuario perciba el cambio.
- Soporte en el XLO de tráfico cifrado.

Parte V

Anexos

Apéndice A

Planificación y presupuesto

En esta sección se va a presentar la descomposición de tareas que se han llevado a cabo para realizar este Trabajo Fin de Grado. Se describen cada una de las tareas así como la relación con otras tareas y el esfuerzo de cada una.

A.1. Descomposición en tareas.

- **Tarea A: Análisis del proyecto y de la demostración.**
  - **Subtarea A.1: Estudio de la documentación de MEDIEVAL**
    - **Descripción:** se estudió la propuesta de MEDIEVAL con la documentación oficial del proyecto.
    - **Objetivos:** comprensión de la propuesta de MEDIEVAL
    - **Dependencia:** da comienzo a este Trabajo de Fin de Grado.
    - **Duración:** 1 semana.
    - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes
  - **Subtarea A.2: Análisis de la demostración 3**
    - **Descripción:** se estudió el objetivo de dicha demostración.
    - **Objetivos:** comprensión de la demostración 3 dentro del proyecto MEDIEVAL.
    - **Dependencia:** da comienzo tras la tarea A.1.
    - **Duración:** 1 semana.
    - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes.
- **Tarea B: Montaje del testbed.**
  - **Tarea B.1: Instalación del sistema operativo en todos los equipos.**
    - **Descripción:** se instaló y configuró el sistema operativo Ubuntu 10.04 LTS en todos los elementos de nuestra topología.
    - **Objetivos:** tener operativos los equipos para el testbed.
    - **Dependencia:** da comienzo tras la tarea A.2.
    - **Duración:** 1 semana.
    - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes

- **Tarea B.2: Configuración de la topología de red.**
  - **Descripción:** se configuró la red de los equipos.
  - **Objetivos:** tener los equipos interconectados según la topología de la demostración.
  - **Dependencia:** da comienzo tras la tarea B.1.
  - **Duración:** 1 semana.
  - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes

- **Tarea C: Configuración e integración de los módulos existentes.**
  - **Tarea C.1: Instalación de los módulos existentes.**
    - **Descripción:** se sincronizó el repositorio subversion con el del proyecto y se instalaron los módulos ya desarrollados.
    - **Objetivos:** tener todas las herramientras listas para probar.
    - **Dependencia:** da comienzo tras la tarea B.2.
    - **Duración:** 1 semana.
    - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes
  - **Tarea C.2: Integración y prueba de los módulos existentes.**
    - **Descripción:** se integraron los módulos disponibles en el repositorio y se probó su funcionamiento.
    - **Objetivos:** comprobar que el comportamiento de las herramientas elegidas es el esperado.
    - **Dependencia:** da comienzo tras la tarea C.1.
    - **Duración:** 1 semana.
    - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes

- **Tarea D: Desarrollo e integración de AirTime.**
  - **Tarea D.1: Diseño de AirTime.**
    - **Descripción:** se diseñó la implementación de la herramienta AirTime.
    - **Objetivos:** desarrollar una herramienta para detectar la congestión en medios inalámbricos basados en IEEE 802.11.
    - **Dependencia:** da comienzo tras la tarea C.2.
    - **Duración:** 1 semana.
    - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes; Ingeniero Senior 0.25 ingenieros/mes.
  - **Tarea D.2: Implementación e integración con el XLO.**
    - **Descripción:** se implementó AirTime en C y se integró con el XLO.
    - **Objetivos:** dotar al XLO de un detector de congestión.
    - **Dependencia:** da comienzo tras la tarea D.1.
    - **Duración:** 1 semana.
    - **Recursos:** Ingeniero 0.25 ingenieros/mes

- **Tarea E: Pruebas de validación.**
  - **Tarea E.1: Validación de los módulos iniciales**
    - **Descripción:** se probaron de nuevo los módulos iniciales para probar el comportamiento de estos.

- **Objetivos:** verificar el comportamiento de los módulos iniciales.
- **Dependencia:** da comienzo tras la tarea D.2.
- **Duración:** 1 semana.
- **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes; Ingeniero Senior 0.25 ingenieros/mes.

- **Tarea E.2: Validación de AirTime y XLO.**
  - **Descripción:** se hicieron diversas pruebas para probar la integración de AirTime junto al XLO.
  - **Objetivos:** verificar que AirTime y XLO trabajan bien conjuntamente.
  - **Dependencia:** da comienzo tras la tarea E.1.
  - **Duración:** 1 semana.
  - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes; Ingeniero Senior 0.25 ingenieros/mes.
- **Tarea E.3: Ajuste de parámetros.**
  - **Descripción:** se hicieron varias pruebas para obtener los umbrales de congestión así como los valores óptimos de las diversas configuraciones.
  - **Objetivos:** calcular los umbrales de congestión para los diferentes escenarios de la demostración.
  - **Dependencia:** da comienzo tras la tarea E.2.
  - **Duración:** 1 semana.
  - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes; Ingeniero Senior 0.25 ingenieros/mes.

- **Tarea F: Mediciones y pruebas**
  - **Tarea F.1: Pruebas de la demostración completa.**
    - **Descripción:** se probaron los diferentes escenarios de la demostración al completo.
    - **Objetivos:** verificar el funcionamiento de todos los escenarios de la demostración.
    - **Dependencia:** da comienzo tras la tarea E.3.
    - **Duración:** 2 semanas.
    - **Recursos:** G. Ingeniería Telemática 0.5 ingenieros/mes; Ingeniero Senior 0.5 ingenieros/mes.
  - **Tarea F.2: Mediciones para cada uno de los escenarios.**
    - **Descripción:** se medió el *bitrate* del vídeo y la QoE del usuario en los diferentes escenarios de la demostración.
    - **Objetivos:** obtener gráficas de la variación del *bitrate* así como la QoE del usuario.
    - **Dependencia:** da comienzo tras la tarea F.1.
    - **Duración:** 1 semana.
    - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes; Ingeniero Senior 0.25 ingenieros/mes.
- **Tarea G: Migración del prototipo al testbed remoto**
  - **Tarea G.1: Migrar la configuración y los módulos al testbed remoto.**

- **Descripción:** se copiaron las configuraciones y módulos del testbed ubicado en la UC3M al de los laboratorios de Eurecom
- **Objetivos:** replicar el testbed de la UC3M.
- **Dependencia:** da comienzo tras la tarea F.2.
- **Duración:** 1 semanas.
- **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes

- **Tarea G.2: Configuración del testbed.**
  - **Descripción:** se adaptó la configuración del testbed de la UC3M al de Eurecom.
  - **Objetivos:** tener listo el testbed remoto para la demostración.
  - **Dependencia:** da comienzo tras la tarea G.1.
  - **Duración:** 1 semana.
  - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes

- **Tarea H: Mediciones y pruebas en el testbed remoto.**
  - **Tarea H.1: Repetir las pruebas realizadas en la UC3M.**
    - **Descripción:** se repitieron las pruebas realizadas en la UC3M en los laboratorios de Eurecom.
    - **Objetivos:** verificar que el funcionamiento y comportamiento de la demostración era el mismo que en el obtenido en la UC3M.
    - **Dependencia:** da comienzo tras la tarea G.2.
    - **Duración:** 1 semanas.
    - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes; Ingeniero Senior 0.25 ingenieros/mes.
  - **Tarea H.2: Medir y comparar los resultados con los datos obtenidos anteriormente.**
    - **Descripción:** se midió de nuevo el *bitrate* y se verificó que la adaptación era la misma que en las pruebas realizadas en la UC3M.
    - **Objetivos:** comprobar que los resultados son consistentes entre los testbed.
    - **Dependencia:** da comienzo tras la tarea H.1.
    - **Duración:** 1 semana.
    - **Recursos:** G. Ingeniería Telemática 0.25 ingenieros/mes; Ingeniero Senior 0.25 ingenieros/mes.

- **Tarea I: Memoria.**
  - **Tarea I.1: Redacción de la memoria.**
    - **Descripción:** se redactó esta memoria.
    - **Objetivos:** obtener la memoria del Trabajo de Fin de Grado.
    - **Dependencia:** da comienzo tras la tarea I.2.
    - **Duración:** 48 días.
    - **Recursos:** G. Ingeniería Telemática 0.75 ingenieros/mes

Tarea	Duración (semanas)	G. Ing. Tlm. (Ing/m)	Ing. Senior (Ing/m)
Análisis del proyecto y de la demostración			
A.1 Estudio de la documentación de MEDIEVAL	1	0.25	-
A.2 Análisis de la demostración 3	1	0.25	-
Total		**0.5**	-
Montaje del testbed			
B.1 Instalación del sistema operativo en todos los eq...	1	0.25	-
B.2 Configuración de la topología de red	1	0.25	-
Total		**0.5**	-
Configuración e integración de los módulos...			
C.1 Instalación de los módulos existentes	1	0.25	-
C.2 Integración y prueba de los módulos existentes	1	0.25	-
Total		**0.5**	-
Desarrollo e integración de AirTime			
D.1 Diseño de AirTime	1	0.25	0.25
D.2 Implementación e integración con XLO	1	0.25	-
Total		**0.5**	**0.25**
Pruebas de validación			
E.1 Validación de los módulos iniciales	1	0.25	0.25
E.2 Validación de AirTime y XLO	1	0.25	0.25
E.3 Ajuste de parámetros	1	0.25	0.25
Total		**0.75**	**0.75**
Mediciones y pruebas			
F.1 Pruebas de la demostración completa	2	0.5	0.5
F.2 Mediciones para cada uno de los escenarios	1	0.25	0.25
Total		**0.75**	**0.75**
Migración del prototipo al testbed remoto			
G.1 Migrar la configuración y los módulos al testbed...	1	0.25	-
G.2 Configuración del testbed	1	0.25	-
Total		**0.5**	-
Mediciones y pruebas en el testbed remoto			
H.1 Repetir las pruebas realizadas en la UC3M	1	0.25	0.25
H.2 Medir y comprar los resultados con los datos...	1	0.25	0.25
Total		**0.5**	**0.5**
Memoria			
I.1 Redacción de la memoria	9.6	0.75	-
Total		**0.75**	-
Total	27.6	**5.25**	**2.25**

Tabla A.1: Resumen descomposición en tareas

A.2. Planificación con el diagrama de fases de ejecución detallado

En la figura A.1 se muestra el diagrama de Gantt con las tareas principales. Además en la figura A.2 se presenta el diagrama de Gantt con todas las tareas que se han realizado

en este Trabajo Fin de Grado.

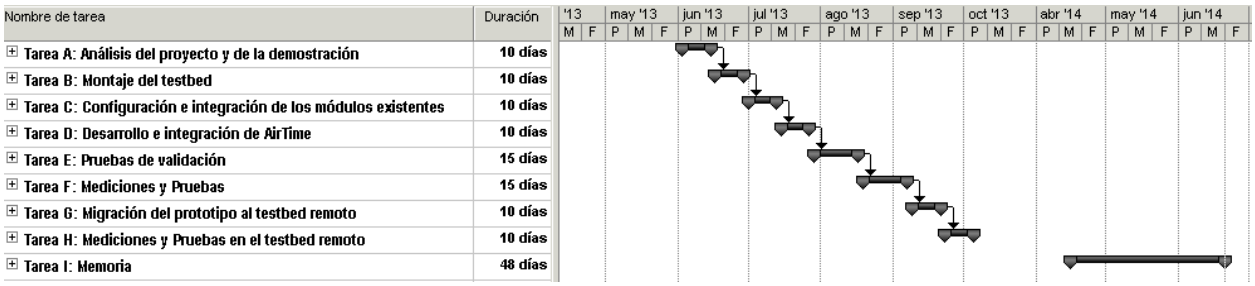

Figura A.1: Diagrama de Gantt con la planificación del proyecto resumida

Nombre de tarea	Duración
Tarea A: Análisis del proyecto y de la demostración	**10 días**
A.1: Estudio de la documentación de MEDIEVAL	1 sem
A.2: Análisis de la demostración	1 sem
Tarea B: Montaje del testbed	**10 días**
B.1: Instalación del sistema operativo en todos los equipos	1 sem
B.2: Configuración de la topología de red	1 sem
Tarea C: Configuración e integración de los módulos existentes	**10 días**
C.1: Instalación de los módulos existentes	1 sem
C.2: Integración y prueba de los módulos existentes	1 sem
Tarea D: Desarrollo e integración de AirTime	**10 días**
D.1: Diseño de AirTime	1 sem
D.2: Implementación e integración con XLO	1 sem
Tarea E: Pruebas de validación	**15 días**
E.1: Validación de los módulos iniciales	1 sem
E.2: Validación de AirTime y XLO	1 sem
E.3: Ajuste de parámetros	1 sem
Tarea F: Mediciones y Pruebas	**15 días**
F.1: Pruebas de la demostración completa	2 sem.
F.2: Mediciones para cada uno de los escenarios	1 sem
Tarea G: Migración del prototipo al testbed remoto	**10 días**
G.1: Migrar la configuración y módulos en el testbed remoto	1 sem
G.2: Configuración del testbed remoto	1 sem
Tarea H: Mediciones y Pruebas en el testbed remoto	**10 días**
H.1: Repetir las pruebas realizadas en la UC3M	1 sem
H.2: Medir y comparar los resultados con los datos obtenidos anteriorm	1 sem
Tarea I: Memoria	**48 días**
I.1: Redacción de la memoria	48 días

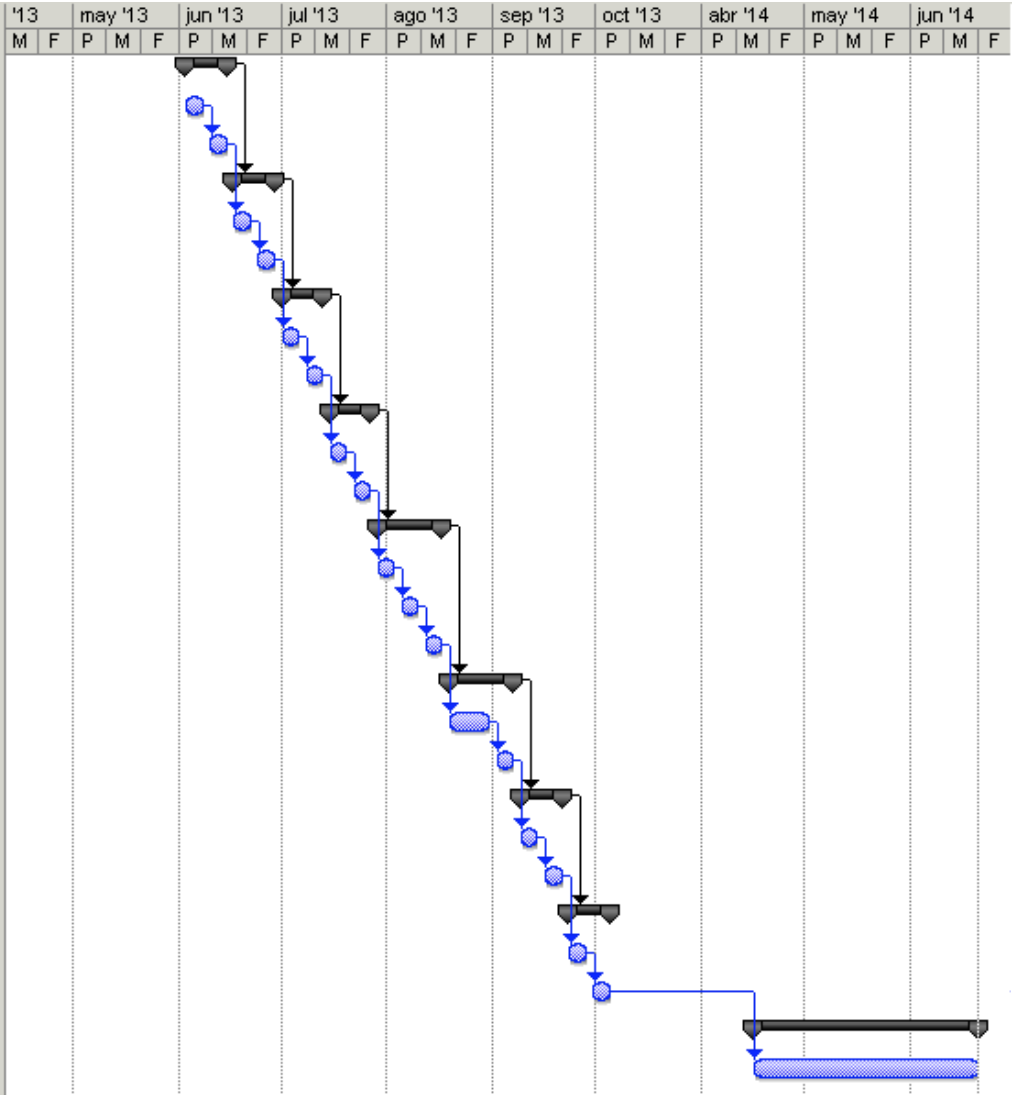

Figura A.2: Diagrama de Gantt con la planificación detallada del proyecto

A.3. Recursos

En esta sección se distinguen los diferentes recursos usados para la realización del Trabajo Fin de Grado:

- Recursos materiales:
  - 4 PCs Intel Core 2 Quad Processor Q9400, 4GB RAM; Sistema Operativo: Ubuntu LTS 10.04
  - 2 PCs portátiles Dell D630 T7500, 2GB RAM; Sistema Operativo: Ubuntu LTS 10.04
  - 5 tarjetas inalámbricas basadas en un chipset Atheros.
- Mano de obra directa:
  - 1 Graduado en Ingeniería Telemática: 5.25 ingenieros/mes
  - 2 Ingenieros Senior: 2.25 ingenieros/mes

A.4. Presupuesto de Proyecto

1. Autor: Carlos A. Donato Morales
2. Departamento: Ingeniería Telemática
3. Descripción del Proyecto:
   - Título: Mecanismos de optimización cross-layer para el tráfico de vídeo en redes 4G.
   - Duración: 4 meses
   - Tasa de costes indirectos: 25 %.
4. Presupuesto total del Proyecto (valorado en Euros): euros. Ver tabla A.2
5. Subcontratación de tareas: no se especifican.
6. Otros costes directos del proyecto: no se especifican.

Concepto	**Cantidad (€)**	**Coste €**	**% Proyecto**	**Dedicación (meses)**	**Depreciación (meses)**	**Total €**
Recursos materiales						
Ordenadores de sobremesa	4	750	100	4	60	200
Ordenadores portátiles	2	950	100	4	60	126,7
Tarjetas de red inalámbricas	5	50	100	4	60	16,67
Total						**343,34**
Mano de obra directa						
Graduado en Ing. Telemática	1 (5.25 ing/mes)	2.694,39	-	-	-	14.145,55
Ingenieros Senior	2 (2.25 ing/mes)	4.289,54	-	-	-	9.651,47
Total						**23.797,02**
Total de costes directos						**24.140,35€**
Costes indirectos						
Costes indirectos	-	-	-	-	-	25 %
Total						**6.035,09**
Total						**30.175,43 €**

Tabla A.2: Tabla de presupuesto

Apéndice B

Configuración de los nodos móviles MN2, MN5 y MN6

B.1. Introducción

En este apéndice se describen las herramientas y configuraciones necesarias para la recepción y reproducción del stream de vídeo del nodo móvil MN2 y MN5 así como para la movilidad de dominios MEDIEVAL y fuera de él. También se detallará la configuración del equipo MN5 para generar la congestión necesaria para demostración.

B.2. Características de los equipos

La configuración hardware y de sistemas operativo de MN2 fue:

- CPU: Core 2 Quad Q9400
- RAM: 4096 MB
- Tarjeta de red: basadas en chipset Atheros.
- Sistema Operativo: Ubuntu 10.04 LTS de 32 bits.

Respecto a MN5 y MN7 se usaron dos equipos portátiles Dell D630 con la siguiente configuración:

- CPU: Core 2 Duo T7500
- RAM: 2048 MB
- Tarjeta de red: basadas en chipset Atheros.
- Sistema Operativo: Ubuntu 10.04 LTS de 32 bits.

B.3. Configuración de MN2

B.3.1. Configuración de las librerías Live555 y del reproductor MPlayer

Para la visualización del vídeo codificado con el códec SVC es necesaria la instalación de las librerías adicionales Live555 y la configuración y compilación del reproductor MPlayer con soporte para dicho códec

B.3.1.1. Librerías Live555

Para instalar y configurar las librerías se siguieron los siguientes pasos:

1. Obtenemos las fuentes desde el sitio web y descomprimimos:

```
wget http://live555.com/liveMedia/public/live555-latest.tar.gz
tar -xzvf live555-lastest.tar.gz
```

2. Compilamos el código fuente e instalamos:

```
cd live555
make -j3
cp -r ../live555 /usr/lib/
```

Con estos pasos se tienen compiladas e instaladas las librerías adicionales para posteriorme poder configurar la compilación de MPlayer con soporte para el códec SVC.

B.3.1.2. Reproductor multimedia MPlayer

1. Obtenemos las fuentes y descomprimimos:

```
wget http://www.mplayerhq.hu/MPlayer/releases/MPlayer-1.1.1.tar.xz
tar -xvzf MPlayer-1.1.1.tar.xz
```

2. Configuramos la compilación para el soporte del códec SVC y compilamos:

```
cd MPlayer-1.1.1
./configure -enable-svc
make -j3
```

Para este caso, no hizo falta instalar el programa puesto que se ejecutaba directamente desde el directorio donde se compiló pero si se quiere instalar se haría con:

```
make install
```

Con esto tendríamos configurado la visualización del stream de vídeo.

B.3.2. Configuración de las tarjetas de red inalámbricas

Siguiendo la topología propuesta, configuraremos los diferentes interfaces de red.

1. Levantamos los interfaces WLAN:

   ```
   ip link set wlan0 up
   ip link set wlan1 up
   ```

   Nota: el identificador del interfaz `wlanX` (siendo X el identificador) puede variar dependiendo de cómo el sistema lo asigne.

2. Conectamos cada interfaz a las diferentes ESSID provistas por MAR3 y MAR6:

   ```
   iwconfig wlan0 essid mar3 ap a0:f3:c1:1c:08:5f rate 54M channel 5
   iwconfig wlan1 essid not-medieval ap 90:f6:52:0c:3b:1b rate 54M
   channel 11
   ```

   Nota: no es necesario especificar la dirección MAC del AP pero se incluyó para aumentar la fiabilidad del comando a la hora de establecer la conexión.

3. Asignamos direcciones IP a los interfaces:

   ```
   ip -6 addr add 2002::2 dev lo
   ip -6 addr add 7000::3/64 dev wlan1
   ```

   En este caso se ha asignado la dirección IP 2002::2 al interfaz *loopback* `lo` puesto que se empleará dicho interfaz como puente entre el interfaz creado por OpenVPN `tap0` y el interfaz físico `wlan0`.

4. Añadimos las rutas iniciales:

   ```
   ip -6 route add 6000::102 via fe80::a2f3:c1ff:fe1c:85f dev wlan0 src
   2002::2
   ```

5. Habilitamos el reenvío de paquetes IPv6 de nuestros interfaces:

   ```
   sysctl -w net.ipv6.conf.wlan0.forwarding=1
   sysctl -w net.ipv6.conf.tap0.forwarding=1
   ```

B.3.3. Configuración del túnel IPv6-IPv4

El servidor de vídeo tiene la limitación de que sólo funciona sobre IPv4 pero nuestra topología está configurada sobre IPv6. Solventamos este problema creando un túnel IPv6-IPv4 entre el servidor de vídeo y el nodo móvil.

1. Creamos el túnel en modo IPv4-IPv6 y levantamos el interfaz creado:

```
ip -6 tunnel add mode ipip6 local 2001:1::3 remote 6000::102 name
VIDEOtun0
ip -6 link set VIDEOtun0 up
```

2. Asignamos una dirección IPv4 a nuestro extremo del túnel:

```
ip addr add 10.0.0.2/24 dev VIDEOtun0
```

3. Modificamos la MTU del túnel puesto que el túnel añade 8 bytes nuevos de cabecera. Con esto evitaremos fragmentación IP.

```
ip link set VIDEOtun0 mtu 1452
```

B.3.4. Configuración de OpenVPN

Para la solución de movilidad, empleamos OpenVPN como pasarela entre MN2 y MAR3. Explicaremos como instalar y configurar el cliente de OpenVPN para nuestro escenario. [ope]

1. Instalamos el paquete de OpenVPN:

```
apt-get install openvpn
```

2. Modificamos el fichero de configuración `client.conf` alojado (habitualmente) en `/etc/openvpn`.

   2.1 Especificamos que este terminal es un cliente:

```
# Specify that we are a client and that we
# will be pulling certain config file directives
# from the server.
client
```

   2.2 Elegimos el prefijo para el interfaz creado por el cliente OpenVPN. Nosotros trabajamos con `tapX`:

```
# Use the same setting as you are using on
# the server.
# On most systems, the VPN will not function
# unless you partially or fully disable
# the firewall for the TUN/TAP interface.
dev tap
```

2.3 Configuramos la conexión en modo UDP para IPv6:

```
# Are we connecting to a TCP or
# UDP server? Use the same setting as
# on the server.
proto udp6
```

2.4 Especificamos la dirección IPv6 del servidor, en nuestro caso, nos conectamos a `5000::53`:

```
# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote 5000::53 1194
```

2.5 Y, para finalizar, debemos añadir el nombre de los ficheros que contienen las claves para el cifrado SSL/TLS que emplea OpenVPN asegurar la conexión:

```
# SSL/TLS parms.
# See the server config file for more
# description. It's best to use
# a separate .crt/.key file pair
# for each client. A single ca
# file can be used for all clients.
ca ca.crt
cert client.crt
key client.key
```

Nota: los ficheros: `ca.crt`, `client.crt` y `client.key` son generados y provistos por el servidor OpenVPN en el momento de su configuración.

B.3.5. Connection Manager

Al igual que el módulo Flow Manager 4.4.3, depende del framework ODTONE para el soporte del protocolo IEEE 802.21. Los pasos a seguir con iguales: instalar primero las librerías ?`boost` y después el paquete de ODTONE.

B.3.5.1. Librerías Boost

Instalamos las librerías desde los repositorios:

```
apt-get install libboost-all-dev
```

B.3.5.2. ODTONE

Editamos el fichero de configuración para ODTONE/Connection Manager `connectionmanager_demo3.conf`:

- Indicamos que vamos a usar el Connection Manager para la demostración 3:

```
[ user ]
id = connectionmanager_demo3
```

- Habilitamos la interfaz gráfica para Connection Manager:

```
[ cmgui ]
gui_enabled = true
```

- Especificamos los interfaces de red a usar para la movilidad:

```
[ interfaces ]
current = WLAN1
next = WLAN2, WLAN1
```

- Incluimos los scripts a ejecutar:

```
[ shell_commands ]
lte_start = /medieval/demo3/mn2/scripts/vpn_lte_mn2.sh
wifi_start = /medieval/demo3/mn2/scripts/lte_vpn_mn2.sh
```

Por último, ejecutaríamos el Connection Manager con:
`./cmgui`

B.4. Configuración de MN5

La configuración de MN5 es, esencialmente, la misma en cuanto a la instalación del reproductor de medios MPlayer y librerías Live555 por lo que se recomienda seguir los pasos anteriormente detallados para el nodo móvil MN2.

B.4.1. Instalación de la herramienta Iperf

`Iperf` es una herramienta para medir el rendimiento del ancho de banda de una red sobre TCP o UDP por lo que es ideal para generar tráfico y, con ello, una congestión en la red.

1. Instalamos el paquete desde los repositorios:

```
apt-get install iperf
```

B.4.2. Configuración de la tarjeta inalámbrica

En este nodo móvil sólo tenemos un interfaz de red inalámbrico asi que procedemos a levantar el interfaz, conectarnos a la ESSID provista por MAR3, asignarle una dirección

IPv6 y ruta inicial.

1. Levantamos el interfaz de red inalámbrico:

   ```
   ip link set wlan0 up
   ```

2. Enlazamos el interfaz a la red inalámbrica de MAR3:

   ```
   iwconfig wlan0 essid mar3 ap a0:f3:c1:1c:08:5f rate 54M channel 5
   ```

3. Asignamos la dirección IPv6 a dicho interfaz:

   ```
   ip -6 addr add 2001:1::5/64 dev wlan0
   ```

4. Añadimos las rutas según la topología:

   ```
   ip -6 route add 2001:1::/64 dev wlan0
   ip -6 route add 6000::/64 via 2001:1::1
   ```

   Nota: suponemos que el interfaz de red inalámbrico creado por sistema es `wlan0` como es nuestro caso.

B.4.3. Servidor Iperf

Para el escenario en el que MN5 congestiona a MN2 en el dominio MEDIEVAL, lanzaremos el servidor `Iperf` en modo UDP ya que este protocolo de transporte es *best-effort* y no tiene ningún mecanismo de adaptación de red como puede ser TCP, haciéndolo idóneo para congestionar nuestra red. Más información en [ipe].

Lo lanzaremos con:

```
iperf -s -V -u
```

B.5. Configuración de MN6

Para este nodo móvil, la configuración es muy sencilla puesto que es solo un equipo destinado a generar congestión por lo que se puede resumir en conectarse a la red con dominio MEDIEVAL y sin él.

B.5.1. Instalación de la herramienta Iperf

`Iperf` es una herramienta para medir el rendimiento del ancho de banda de una red sobre TCP o UDP por lo que es ideal para generar tráfico y, con ello, una congestión en la red.

1. Instalamos el paquete desde los repositorios:

```
apt-get install iperf
```

B.5.2. Configuración de las librerías Live555 y del reproductor MPlayer

Para la visualización del vídeo codificado con el códec SVC es necesaria la instalación de las librerías adicionales Live555 y la configuración y compilación del reproductor MPlayer con soporte para dicho códec

B.5.2.1. Librerías Live555

Para instalar y configurar las librerías se siguieron los siguientes pasos:

1. Obtenemos las fuentes desde el sitio web y descomprimimos:

```
wget http://live555.com/liveMedia/public/live555-latest.tar.gz
tar -xzvf live555-lastest.tar.gz
```

2. Compilamos el código fuente e instalamos:

```
cd live555
make -j3
cp -r ../live555 /usr/lib/
```

Con estos pasos se tienen compiladas e instaladas las librerías adicionales para posteriorme poder configurar la compilación de MPlayer con soporte para el códec SVC.

B.5.2.2. Reproductor de medios MPlayer

1. Obtenemos las fuentes y descomprimimos:

```
wget http://www.mplayerhq.hu/MPlayer/releases/MPlayer-1.1.1.tar.xz
tar -xvzf MPlayer-1.1.1.tar.xz
```

2. Configuramos la compilación para el soporte del códec SVC y compilamos:

```
cd MPlayer-1.1.1
./configure -enable-svc
make -j3
```

Para este caso, no hizo falta instalar el programa puesto que se ejecutaba directamente desde el directorio donde se compiló pero si se quiere instalar se haría con:

```
make install
```

Con esto tendríamos configurado la visualización del stream de vídeo.

B.5.3. Configuración de la tarjeta inalámbrica

Como en secciones anteriores, explicaremos tanto la configuración necesaria de la tarjeta inalámbrica para conectarnos a la red de dominio MEDIEVAL y cómo movernos de red.

1. Levantamos el interfaz de red inalámbrico:

   ```
   ip link set up dev wlan0
   ```

2. Enlazamos el interfaz a la red inalámbrica de MAR3:

   ```
   iwconfig wlan0 essid mar3 ap a0:f3:c1:1c:08:5f rate 54M channel 5
   ```

3. Asignamos las direcciones IPv6 del dominio MEDIEVAL y dominio No-MEDIEVAL a dicho interfaz:

   ```
   ip -6 addr add 2001:1::7/64 dev wlan0
   ip -6 addr add 7000::7/64 dev wlan0
   ```

4. Añadimos las rutas por defecto para ambos dominios:

   ```
   ip -6 route add 2001:1::/64 dev wlan0
   ip -6 route add 7000::/64 dev wlan0
   ```

5. Para movernos al dominio No-MEDIEVAL, nos conectaríamos a MAR6:

   ```
   iwconfig wlan0 essid not-medieval ap 90:f6:52:0c:3b:1b rate 54M
   channel 11
   ```

B.5.4. Configuración del túnel IPv6-IPv4

El servidor de vídeo tiene la limitación de que sólo funciona sobre IPv4 pero nuestra topología está configurada sobre IPv6. Solventamos este problema creando un túnel IPv6-IPv4 entre el servidor de vídeo y el nodo móvil.

1. Creamos el túnel en modo IPv4-IPv6 y levantamos el interfaz creado:

   ```
   ip -6 tunnel add mode ipip6 local 2001:1::7 remote 6000::102 name
   VIDEOtun0
   ip -6 link set VIDEOtun0 up
   ```

2. Asignamos una dirección IPv4 a nuestro extremo del túnel:

   ```
   ip addr add 10.0.1.2/24 dev VIDEOtun0
   ```

3. Modificamos la MTU del túnel puesto que el túnel añade 8 bytes nuevos de cabecera. Con esto evitaremos fragmentación IP.

   ```
   ip link set VIDEOtun0 mtu 1452
   ```

B.5.5. Servidor Iperf

Para el escenario en el que MN6 congestiona a MN2 en el dominio no-MEDIEVAL, lanzaremos el servidor `Iperf` en modo UDP ya que este protocolo de transporte es *best-effort* y no tiene ningún mecanismo de adaptación de red como puede ser TCP, haciéndolo idóneo para congestionar nuestra red.

Mas información en [ipe].

Lo lanzaremos con:

```
iperf -s -V -u
```

Apéndice C

Configuración de los router de acceso móviles MAR3 y MAR6

C.1. Introducción

En este apéndice se describen las herramientas y configuraciones necesarias del router de acceso móviles MAR3 para que efectue como *router*, servidor de OpenVPN y sea capaz de detectar congestión en tiempo real. A su vez, se configurará MAR6 como router que dará cobertura a la red con dominio No-MEDIEVAL.

C.2. Características de los equipos

La configuración hardware de ambos MAR es la de un equipo de escritorio con los siguientes componentes y sistema operativo:

- CPU: Core 2 Quad Q9400
- RAM: 4096 MB
- Tarjeta de red: basadas en chipset Atheros.
- Sistema Operativo: Ubuntu 10.04 LTS de 32 bits.

C.3. Configuración de MAR3

C.3.1. Instalación y configuración de Hostapd

En esta sección se explicará como instalar y configurar el demonio `hostapd` para que el equipo sea capaz de proveer una red inalámbrica basada en IEEE 802.11 a través de unos de uno de sus interfaces disponibles. Más información en [hos].

1. Instalamos el paquete `hostapd`:

```
apt-get install hostapd
```

2. Una vez instalado, deberemos configurar el fichero de configuración `hostapd.conf` ubicado en `/etc/hostapd/`.

   2.1 Especificamos el interfaz que vamos a usar para proveer la red inalámbrica. En nuestro *testbed*:

   ```
   # AP netdevice name (without 'ap' postfix, i.e., wlan0 uses
   wlan0ap for
   # management frames); ath0 for madwifi
   interface=wlan0
   ```

   2.2 Lo siguiente que necesitamos configurar es el nombre de nuestra *SSID*:

   ```
   # SSID to be used in IEEE 802.11 management frames
   ssid=mar3
   ```

   2.3 Especificamos el modo de operación en función de los soportados por nuestra tarjeta inalámbrica. Para nuestro caso, usamos el modo *g* cuyo ancho de banda máximo teórico es 54 Mbps:

   ```
   # Operation mode (a = IEEE 802.11a, b = IEEE 802.11b, g = IEEE
   802.11g,
   # ad = IEEE 802.11ad (60 GHz); a/g options are used with IEEE
   802.11n, too, to
   # specify band)
   # Default: IEEE 802.11b
   hw_mode=g
   ```

   2.4 Elegimos un canal para dicha red. Es altamente recomendable elegir uno que este lo mas libre posible:

   ```
   # If CONFIG_ACS build option is enabled, the channel can be
   selected
   # automatically at run time by setting channel=acs_survey or
   channel=0, both of
   # which will enable the ACS survey based algorithm.
   channel=5
   ```

   El resto de parámetros los dejamos por defecto puesto que para la demostración no era necesacio emplear ningún tipo de cifrado ni ajuste particular.

C.3.2. Instalación y configuración de OpenVPN

Para la solución de movilidad, empleamos OpenVPN como pasarela entre MAR3 y MN2. Explicaremos como instalar y configurar el cliente de OpenVPN para nuestro escenario. [ope]

1. Instalamos el paquete de OpenVPN:

   ```
   apt-get install openvpn
   ```

2. Modificamos el fichero de configuración `client.conf` alojado (habitualmente) en `/etc/openvpn` junto con los *scripts* de generación de certificados y claves.

   2.1 Elegimos el prefijo para el interfaz creado por el cliente OpenVPN. Nosotros trabajamos con `tapX`:

   ```
   # Use the same setting as you are using on
   # the server.
   # On most systems, the VPN will not function
   # unless you partially or fully disable
   # the firewall for the TUN/TAP interface.
   dev tap
   ```

   2.2 Configuramos la conexión en modo UDP para IPv6:

   ```
   # TCP or UDP server?
   proto udp6
   ```

   2.3 Especificamos que es un servidor IPv6 y la dirección de subred, en nuestro caso, empleamos `2333::/64`:

   ```
   # Configure server mode and supply a VPN subnet
   # for OpenVPN to draw client addresses from.
   # The server will take 10.8.0.1 for itself,
   # the rest will be made available to clients.
   # Each client will be able to reach the server
   # on 10.8.0.1. Comment this line out if you are
   # ethernet bridging. See the man page for more info.
   server-ipv6 2333::/64
   ```

   2.4 Y, para finalizar, debemos añadir el nombre de los ficheros que contienen las claves para el cifrado SSL/TLS que emplea OpenVPN para asegurar la conexión:

   ```
   # SSL/TLS parms.
   # See the server config file for more
   # description. It's best to use
   # a separate .crt/.key file pair
   # for each client. A single ca
   # file can be used for all clients.
   ca ca.crt
   cert server.crt
   key server.key
   ```

   2.5 Generamos los ficheros con las claves tanto para el servidor como para el cliente.

```
./build-ca
./build-key-server server
./build-key client
```

C.3.3. Configuración de las tarjetas de red inalámbricas

Siguiendo la topología propuesta, configuraremos los diferentes interfaces de red.

1. Levantamos el interfaz WLAN donde va a correr `hostapd`:

   ```
   ip link set wlan0 up
   ```

   Nota: el identificador del interfaz wlanX (siendo X el identificador) puede variar dependiendo de cómo el sistema lo asigne.

2. Asignamos dirección IPv6 al interfaz:

   ```
   ip -6 addr add 2001:1::1/64 dev wlan0
   ```

3. Añadimos las rutas iniciales:

   ```
   ip -6 route add 2002::2 via fe80::92f6:52ff:fe0c:329e dev wlan0
   ip -6 route add 7000::/64 via 5000::56
   ip -6 route add 6000::102 via fe80::5ed9:98ff:feb1:4edd dev eth1
   ```

4. Habilitamos el reenvío de paquetes IPv6 de todos los interfaces:

   ```
   sysctl -w net.ipv6.conf.all.forwarding=1
   ```

5. Tiramos el interfaz WLAN donde va a correr AirTime para ponerla posteriormente en modo monitor:

   ```
   ip link set wlan1 down
   ```

6. Configuramos la segunda tarjeta inalámbrica en modo monitor para usarla con AirTime. Especificamos el canal donde vamos auditar, en este caso, es el 5 al que pertenece el dominio MEDIEVAL:

   ```
   iwconfig wlan1 mode monitor channel 5
   ```

7. Levantamos el interfaz WLAN de nuevo:

   ```
   ip link set wlan1 up
   ```

8. Añadimos reglas con el comando `tc` para limitar el interfaz `wlan0` a un ancho de banda máximo de 2.7 Mbps:

```
tc qdisc add dev wlan0 handle 1:0 root dsmark indices 1 default_index
0
tc qdisc add dev wlan0 handle 2:0 parent 1:0 tbf burst 40960 limit
40960 mtu 1514 rate 2700000bps
```

C.3.4. Reglas ip6tables para XLO

El XLO necesita poder tener acceso a los paquetes de vídeo que MAR3 está encaminando cuando empieza el flujo de vídeo. Para ello, hace uso del *firewall* de Linux para IPv6 llamado `ip6tables` que con un par de reglas es capaz de ver los paquetes que MAR3 va a encaminar.

Por lo tanto, añadimos estas reglas:

```
ip6tables -t mangle -A PREROUTING -d 2002::2/64 -s 6000::102/64 -j NFQUEUE
-queue-num 1
ip6tables -t mangle -A PREROUTING -d 2001:1::5/64 -s 6000::102/64 -j
NFQUEUE -queue-num 1
```

Con estas reglas, el XLO puede acceder a los paquetes cuyo origen sea Server2 (`6000::102/64`), destino sea el interfaz del propio MAR3 (`2002::2/64`) que tiene que encaminar el paquete al siguiente salto que es MN2 (`2001:1::5/64`).

C.3.5. Intalación y configuración de Flow Manager

Este módulo está basado en el framework ODTONE que, a su vez, depende de las librerías adicionales de C++ `boost`. Aquí se describen los pasos a seguir para la correcta instalación.

C.3.5.1. Instalación de las librerías Boost

Instalamos las librerías desde los repositorios:

```
apt-get install libboost-all-dev
```

C.3.5.2. Configuración de ODTONE

Para la configuración de ODTONE, editamos el fichero de configuración `flowmanager.conf`:

- Indicamos el cliente que va a usar ODTONE:

```
##
## User id
##
[ user ]
id = connectionmanager_demo3
```

- Añadimos los interfaces a usar:

```
[ iface_wifi ]
id = wlan1
mar = mihf_mar3
id = wlan0
mar = mihf_mar3_2
```

- Indicamos los scripts a ejecutar para la movilidad:

```
[ shell_commands ]
lte_start = /medieval/demo3/mn2/script/vpn_lte_mn2.sh
umts_start = /medieval/demo3/mn2/script/lte_vpn_mn2.sh
```

Solo quedaría ejecutar ODTONE junto al Flow Manager con: `./odtone-mihf`

C.3.6. AirTime

La utilidad desarrollada para medir el tiempo de uso en el aire o tiempo que emplea una trama IEEE 802.11 en ser transmitida tiene como dependencia la librería `libpcap`. Está librería permite acceder a los paquetes recibidos en un interfaz de red y subirlo a la aplicación en *crudo*, es decir, en su totalidad sin ser desencapsulado por cada uno de los niveles de la pila. Es una librería totalmente probada y que varíos *sniffers* de red se basan en ella como `tcpdump` o `Wireshark`.

1. Para instalar dicha librería:

```
apt-get install libpcap-dev
```

2. Compilamos AirTime:

```
make
```

3. Ejecutamos AirTime sobre el interfaz de inalámbrico que previamente hemos configurado en modo monitor. También deberemos especificar el tiempo sobre cada cuánto medir y reportar (en milisegundos) y entre qué umbrales queremos detectar medios y altos niveles de congestión (en milisegundos también):

```
./airtime wlan1 500 10 22
```

En este caso, medimos y reportamos al XLO cada 500 ms las medidas recogidas e indicamos si ha sobrepasado los umbrales que hemos prefijado al principio de la prueba.

C.3.7. Instalación y configuración de Iperf

Para congestionar la red, lanzaremos un cliente de `iperf` que se conectará al servidor, ya lanzado previamente, en MN7 cuya dirección IPv6 es `2001:1::7/64` en el dominio MEDIEVAL y `7000::7/64` fuera de él.

En este caso, congestionaremos 2 Mbps de los 2.7 disponibles puesto que redujimos el ancho de banda máximo del interfaz inalámbrico de MAR3. Además, hemos especificado que lo haga durante 20 segundos para visualizar la detección de congestión, escalar la calidad del visionado y ver como, cuando termine `iperf`, se recupere y vuelva al estado inicial. La congestión la variaremos en función de la velocidad. Más información en [ipe].

Lo instalaremos con:

```
apt-get install iperf
```

Lo lanzaremos con:

```
iperf -c 2001:1::7 -V -u -b 2Mbps -t 20 -i 2
```

C.4. Configuración de MAR6

La configuración de MAR6 es bastante sencilla y se resume en configurar `hostapd` para la proveer una red inalámbrica, asignar una dirección IPv6 al interfaz inalámbrico y añadir las rutas necesarias.

C.4.1. Configuración de hostapd

La instalación de `hostapd` es idéntica a los pasos seguidos para MAR3 por lo que se recomienda seguir la sección C.3.1. Más información en [hos].

Para la configuración, editaremos el fichero de configuración `hostapd.conf` ubicado en `/etc/hostapd/`.

1. Especificamos el interfaz que vamos a usar para proveer la red inalámbrica. En nuestro *testbed*:

```
# AP netdevice name (without 'ap' postfix, i.e., wlan0 uses wlan0ap
for
# management frames); ath0 for madwifi
```

```
interface=wlan0
```

2. Lo siguiente que necesitamos configurar es el nombre de nuestra *SSID*:

```
# SSID to be used in IEEE 802.11 management frames
ssid=not-medieval
```

3. Especificamos el modo de operación en función de los soportados por nuestra tarjeta inalámbrica. Para nuestro caso, usamos el modo *g* cuyo ancho de banda máximo teórico son 54 Mbps:

```
# Operation mode (a = IEEE 802.11a, b = IEEE 802.11b, g = IEEE
802.11g,
# ad = IEEE 802.11ad (60 GHz); a/g options are used with IEEE 802.11n,
too, to
# specify band)
# Default: IEEE 802.11b
hw_mode=g
```

4. Elegimos un canal para dicha red distinto al provisto por MAR3 y es recomendable que sea ortogonal para prevenir interferencias entre canales:

```
# If CONFIG_ACS build option is enabled, the channel can be selected
# automatically at run time by setting channel=acs_survey or
channel=0, both of
# which will enable the ACS survey based algorithm.
channel=11
```

El resto de parámetros los dejamos por defecto puesto que para la demostración no era necesacio emplear ningún tipo de cifrado ni ajuste particular.

C.4.2. Configuración de la tarjeta inalámbrica

Siguiendo la topología propuesta, configuraremos los diferentes interfaces de red.

1. Levantamos el interfaz WLAN donde va a correr `hostapd`:

```
ip link set wlan0 up
```

2. Añadimos la dirección IPv6 al interfaz de red:

```
ip -6 addr add 7000::1/64 dev wlan0
```

3. Habilitamos el reenvío de paquetes:

```
sysctl -w net.ipv6.conf.all.forwarding=1
```

4. Añadimos reglas con el comando `tc` para limitar el interfaz `wlan0` a un ancho de banda máximo de 2.7 Mbps:

```
tc qdisc add dev wlan0 handle 1:0 root dsmark indices 1 default_index
0
tc qdisc add dev wlan0 handle 2:0 parent 1:0 tbf burst 40960 limit
40960 mtu 1514 rate 2700000bps
```

C.4.3. Configuración de Iperf

Al igual que sucede con MAR3, tenemos que instalar `iperf` y lanzar el cliente para mostrar qué sucede en entornos donde MEDIEVAL no está funcionando. La instalación está detallada en la propia sección de MAR3. La instalación de `iperf` está detallada en la sección C.3.7. Más información en [ipe].

El cliente lo lanzaremos con los siguientes parámetros:

```
iperf -c 7000::7 -V -u -b 2Mbps -t 20 -i 2
```

Apéndice D

Configuración del servidor de vídeo Server2

D.1. Introducción

En este apéndice se describen las herramientas y configuraciones necesarias para la transmisión de vídeo desde el servidor de vídeo Server2 hasta los nodos móviles.

D.2. Característica del equipo

La configuración hardware de Server2 es la de un equipo de escritorio clónico con los siguientes componentes y sistema operativo:

- CPU: Core 2 Quad Q9400
- RAM: 4096 MB
- Tarjeta de red: basadas en chipset Atheros.
- Sistema Operativo: Ubuntu 10.04 LTS de 32 bits.

D.3. Configuración del equipo

D.3.1. Librerías Live555 y servidor de vídeo

Para instalar y configurar las librerías se siguieron los siguientes pasos al igual que en el nodo móvil MN2:

1. Obtenemos las fuentes desde el sitio web y descomprimimos:

```
wget http://live555.com/liveMedia/public/live555-latest.tar.gz
tar -xzvf live555-lastest.tar.gz
```

2. Compilamos el código fuente e instalamos:

```
cd live555
make -j3
cp -r ../live555 /usr/lib/
```

3. Lanzamos el servidor:
```
cd /usr/lib/live555/testProgs
./testOnDemandRTSPServer_8556foreman
```

Así tendríamos configurado las librerías de transporte Live555 y el servidor de vídeo corriendo.

D.3.2. Configuración del túnel IPv6-IPv4

Dada la limitación del servidor de vídeo por falta de soporte para IPv6, es necesario crear dos túneles IPv6-IPv4 para cada uno de los nodos móviles.

1. Creamos dos túneles en modo IPv4-IPv6 para los nodos móviles MN2 y MN6 y levantamos el interfaz creado:

```
ip -6 tunnel add mode ipip6 local 6000::102 remote 2001:1::3 name
VIDEOtun1
ip -6 link set VIDEOtun1 up
sudo ip -6 tunnel add mode ipip6 local 6000::102 remote 2001:1::5 name
VIDEOtun2
ip -6 link set VIDEOtun2 up
```

2. Asignamos una dirección IPv4 a nuestros extremos del túnel:

```
ip addr add 10.0.0.1/24 dev VIDEOtun1
ip addr add 10.0.1.1/24 dev VIDEOtun2
```

3. Modificamos la MTU del túnel puesto que el túnel añade 8 bytes nuevos de cabecera. Con esto evitaremos fragmentación IP.

```
ip link set VIDEOtun1 mtu 1452
ip link set VIDEOtun2 mtu 1452
```

Bibliografía

[fLman08] IEEE Standard for Local and metropolitan area networks. Part 21: Media Independent Handover Services, 2008.

[FP7] Documento interno FP7. http://www.ict-medieval.eu/.

[GB] L. Grieco and S. Mascolo G. Boggia, P. Camarda. Feedback-based control for providing real-time services with the 802.11e MAC.

[GLD+08] S. Gundavelli, K. Leung, V. Devarapalli, K. Chowdhury, and B. Patil. Proxy Mobile IPv6. RFC 5213, August 2008.

[hos] Hostapd. http://wireless.kernel.org/en/users/Documentation/hostapd.

[Ind] Cisco Visual Networking Index. Global mobile data traffic forecast update. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html.

[ipe] Iperf. http://iperf.fr.

[LB] M. Levorato and M. Zorzi L. Badia, N. Baldo. A Markov framework for error control techniques based on selective retransmission in video transmission over wireless channels.

[LHS] K. Langendoen R. Lagendijk L. Haratcherev, 1. Taal and H. Sips. Optimized video streaming over 802.11 by cross-layer signaling.

[MvdS] S. Choi and X. Xu M. van der Schaar, S. Krishnamachari. Adaptive cross-layer protection strategies for robust scalable video transmission over 802.11 WLANs.

[ope] OpenVPN. http://www.openvpn.net.

[OWS] Sección de noticias del sitio web oficial del proyecto MEDIEVAL. http://www.ict-medieval.eu/4.html, última vez visitada en Junio '14.

[PB] E. Masala E. Filippi and 1. De Martin P. Bucciol, G. Davini. Cross layer perceptual ARQ for H.264 video streaming over 802.11 wireless networks.

[PP] P. Serrano P. Patras, A. Banchs. A Control Theoretic Approach for Throughput Optimization in 802.11e EDCA WLAN.

[PW04] M. Pinson and S. Wolf. A new standardized method for objectively measuring video quality, September 2004.

[SD00] N.K. Kakani and S.K. Sen S.K. Das, R. Jayaram. A call admission and control scheme for quality-of-service (QoS) provisioning in next generation wireless networks, 2000.

[ST99] T. Suzuki and S. Tasaka. Performance evaluation of integrated video and data transmission with the IEEE 802.11 standard MAC protocol, 1999.

[V7.06] 3GPP TR 25.913 V7.3.0. Requirements for Evolved UTRA (E-UTRA) and Evolved UTRAN (E-UTRAN), 2006.